Воронежский государственный университет
Факультет прикладной математики, информатики и механики

Математика, информационные технологии, приложения

Межвузовская научная конференция молодых ученых и студентов

Воронеж,
27 апреля 2020 г.

Воронеж
Издательско-полиграфический центр
«Научная книга»
2020

УДК 531(063)+51-7(063)
ББК 22.2я5+22.1я5
М34

Математика, информационные технологии, приложения : сборник трудов Межвузовской научной конференции молодых ученых и студентов, Воронеж, 27 апреля 2020 г. – Воронеж : Научная книга, 2020. – 330 с.
М34
ISBN 978-5-4446-1406-8

Традиционно конференция является междисциплинарной. Важнейшая ее цель – ознакомление молодых специалистов с новыми веяниями в современной науке, а также обмен знаниями и достижениями в предложенных научных направлениях.

УДК 531(063)+51-7(063)
ББК 22.2я5+22.1я5

ISBN 978-5-4446-1406-8

УДК 519.876.5

МОДЕЛИРОВАНИЕ РАСПРОСТРАНЕНИЯ ЭПИДЕМИИ В ОТДЕЛЬНОЙ ПОПУЛЯЦИИ В СИСТЕМЕ ANYLOGIC

С. В. Агеева, А. А. Кособуцкая

Воронежский государственный университет

В современном мире борьба с инфекционными болезнями представляет собой важную задачу здравоохранения во всех странах. Грипп, коронавирусная инфекция и другие острые респираторные вирусные инфекции (ОРВИ) находятся на первом месте по числу ежегодно заболевающих людей. Несмотря на постоянные усилия, направленные на борьбу с возбудителями гриппа, коронавирусной инфекции и других ОРВИ победить их до сих пор не удается. Ежегодно от осложнений гриппа погибают тысячи человек. Это связано с тем, что вирусы, прежде всего вирусы гриппа и коронавирусы, обладают способностью менять свою структуру и мутировавший вирус, способен поражать человека вновь [1].

Как правило, заражение возникает в результате передачи специфического инфекционного агента или его токсичных продуктов от зараженного человека или животного воспринимаемому хозяину прямым либо не прямым путем. Активное взаимодействие между инфицированными и здоровыми людьми может привести к массовым заражениям. В связи с массовым движением больших групп людей возникают условия для взрывного распространения инфекционных заболеваний [3]. Для предотвращения развития эпидемии необходимо ограничить передвижение и изолировать как источник заражения, так и здоровые особи. Для того чтобы показать опасность распространения инфекций в данной статье смоделированы различные варианты развития эпидемии. Симуляция проведена в системе имитационного моделирования AnyLogic [4].

Для проведения исследования была создана популяция из 100 особей, часть из которых была заражена [2].

Для эксперимента создадим больницу, в которую будут помещаться все зараженные особи (рис. 1). В качестве доступа к зараженным создадим дверь, через которую здоровые особи могут попадать к зараженным. В ходе эксперимента будем менять размеры двери и смотреть, как изменяется ситуация. Предположим, что испытуемые отправляются в больницу сразу после заражения с вероятностью 0.8, а внутри больницы возможно заражение и при не прямом контакте с вероятностью 0.9.

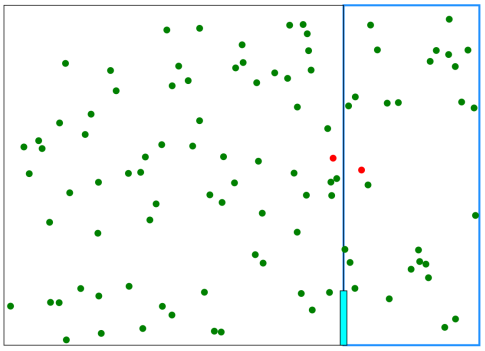
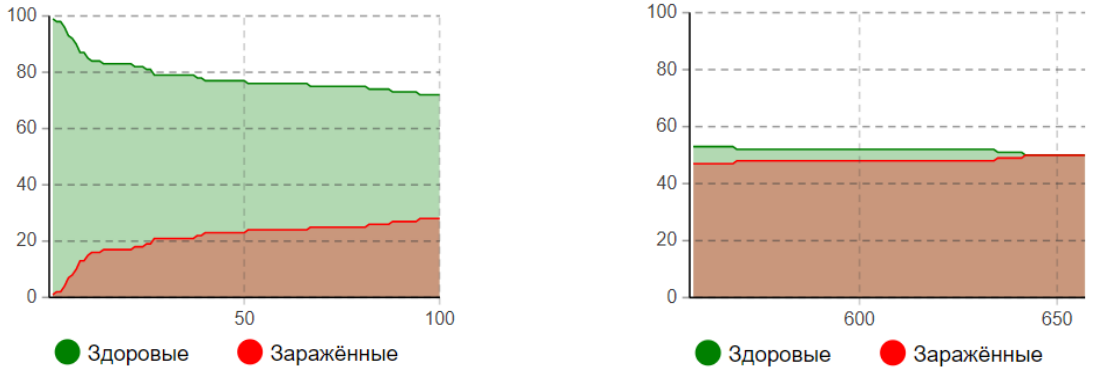


Рис. 1. Популяция. Справа отчерчена область больницы с дверью внизу

Для начала сделаем размер двери 20. В этом случае через 100 условных единиц времени в зоне больницы окажется 28 зараженных особей (рис. 2а).

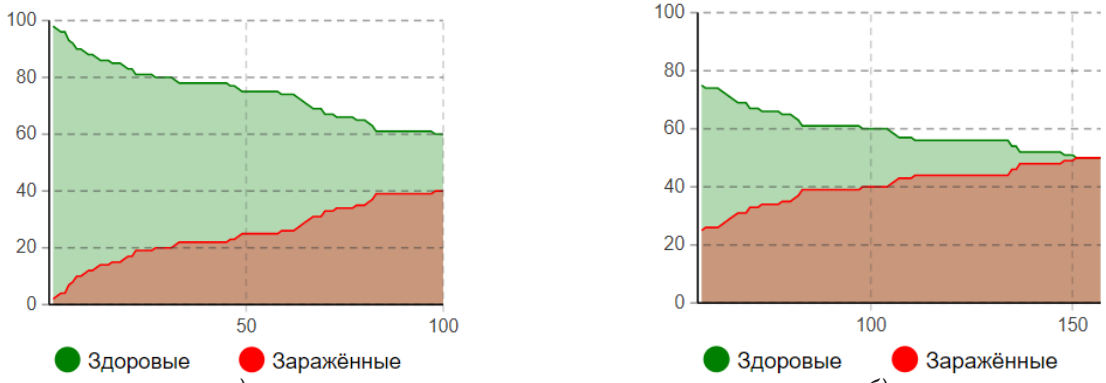


а) *б)*

Рис. 2. График скорости распространения эпидемии при размере двери 20
а) начало заражения, б) момент заражения 50% популяции

На графике видно, что в первые 10 единиц времени происходит резкий рост количества зараженных. Далее следует равномерное течение заболевания с постепенным увеличением больных. В результате чего заражение 50 % популяции произойдет через 642 единицы времени (рис. 2б).

Теперь сделаем размер двери 50. В этом случае заражение происходит с умеренной скоростью. За 100 условных единиц времени заразится 40 особей (рис. 3а), а заражение 50% популяции произойдет через 152 единицы времени (рис. 3б).



а) *б)*

Рис. 3. График скорости распространения эпидемии при размере двери 50
а) начало заражения, б) момент заражения 50% популяции

Если увеличить размер двери до 80, то на графике видно, что на протяжении всего времени происходит быстрое заражение популяции. Уже через 70 условных единиц времени в зоне больницы окажется 56 зараженных, что составляет больше 50 % от всего населения (рис. 4).

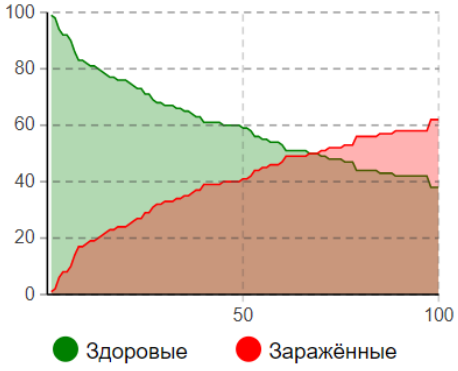


Рис. 4. График скорости распространения эпидемии при размере двери 80

Если же полностью убрать дверь и предоставить возможность здоровым особям беспрепятственно попадать в больницу, то наблюдается резкий скачок роста заболеваемости и уже через 8 единиц времени заразится половина популяции, а через 75 единиц времени останется только 2 здоровые особи (рис. 5).

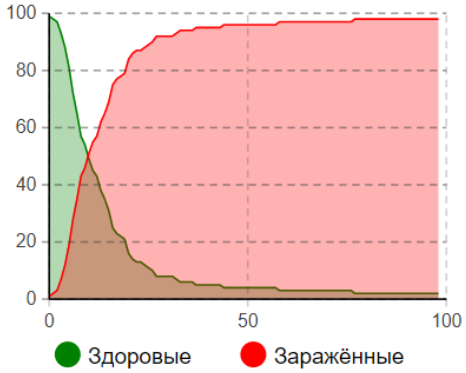


Рис. 5. График скорости распространения эпидемии при свободном доступе в больницу

Заключение

Таким образом, в ходе проведенного исследования удалось смоделировать различные ситуации, исходя из которых можно сделать следующий вывод: для замедления распространения вирусных инфекций необходимо, чтобы максимально большое количество зараженных находилось на изоляции (в больнице) и допуск к ним был ограничен.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Биглхол, Р.* Основы эпидемиологии : [Пер. с англ.] / Р. Биглхол, Р. Бонита, Т. Кьельстрем. – Москва : Медицина, 1994. – 258 с.

2. *Медведев, С. Н.* Имитационное моделирование в среде AnyLogic : учебно-методическое пособие для вузов / О. Г. Корольков, С. Н. Медведев, О. А. Медведева. – Воронеж : ВГУ, 2018. – 74 с.

3. Pastor-Satorras R. and Vespignani A. Epidemic spreading in scale-free networks, Phys. Rev. Lett., Vol. 86. – P. 3200–3203, 2001.

4. AnyLogic : имитационное моделирование для бизнеса. – Режим доступа: https://www.anylogic.ru (Дата обращения: 24.04.2020).

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

Агеева Светлана Владимировна – студентка 2-го курса направления прикладная математика и информатика Воронежского государственного университета.

E-mail: sveta.ageeva.99@mail.ru

Кособуцкая Анна Алексеевна – студентка 2-го курса направления прикладная математика и информатика Воронежского государственного университета.

E-mail: s44ths@yandex.ru

УДК 004.932.2

ВЛИЯНИЕ ПРЕДОБРАБОТКИ ИЗОБРАЖЕНИЙ НА СЕГМЕНТАЦИЮ С ПОМОЩЬЮ НЕЙРОННЫХ СЕТЕЙ

И. Е. Амелин

Воронежский государственный университет

Введение

В прикладных задачах, связанных с обработкой изображениях, часто требуется выделение границ и объектов для последующей работы с ними. Процесс разделения цифрового изображения на несколько групп пикселей – областей, соответствующих по смысле одному объектов с одновременным определением типа объекта, называется сегментацией изображения. В процессе сегментации каждому пикселю изображения присваивается метка, так что пиксели с одинаковыми метками имеют общие визуальные характеристики, а соседние области значительно отличаются друг от друга [1].

Задача сегментации изображений решается с использованием различных подходов, среди которых можно выделить методы, основанные на анализе яркости изображения [2]; методы, использующие для сегментации теорию графов [3]; кластерный анализ [4], нейросетевые технологии [5], причем наилучшие результаты, по мнению исследователей [6], показывают свёрточные нейронные сети.

Заметим, что при использовании нейронных сетей качество решения задачи во многом определяется качеством обучения. Данный этап может оказаться довольно затратным с точки зрения вычислительных ресурсов, поэтому проводится предобработка данных – изменение характеристик изображения с целью получения желаемых свойств: повышения четкости границ объектов на изображении, устранении шумов и т. д.

Цель статьи заключается в исследовании влияния на сегментацию изображения одного из методов предобработки – повышения резкости изображений, как одного из простых и наиболее используемых при работе с изображения.

1. Материалы и методы исследования

1.1. Процедура повышения резкости изображения

Изображения представляются набором пикселей. Каждому пикселю ставится в соответствие тройка чисел $< x^R, x^G, x^B >$, где $x^R, x^G, x^B = 0,1,...,255$ – интенсивность красного, зеленого и синего цветов соответственно. Для более работы с этими значениями изображение представляется тремя матрицами R, G, B размерности $w \times h$, где w – ширина, h – высота изображения в пикселях, элементы которых описывают интенсивность каждого цвета соответствующих пикселей.

Матричное представление позволяет менять изображения определенным образом в зависимости от поставленной задачи. В частности, обработка может использоваться для повышения качества изображения, что в дальнейшем позволяет более эффективно решать другие задачи. Обработка изображений, в результате которой из исходного изображения получается новое, с некоторыми необходимыми свойствами, называется *фильтрацией изображений*, а процедуры фильтрации – *фильтрами* [2, 7].

Различают фильтры *пространственные*, работающие непосредственно с пикселями изображения, и *частотные*, которые имеют дело с частотным спектром изображения. Пространственные фильтры описываются уравнением вида

$$g(x,y) = T\left[f(x,y)\right],$$

где $f(x,y)$ – входное изображение;

$g(x,y)$ – обработанное изображение;

T – оператор над f, определенный в некоторой окрестности каждой точки (x,y), называемый *ядром фильтра* [2].

По сути, во время фильтрации изображения происходит перенос начала координат окрестности от точки к точке с последующим применением оператора T к пикселям этой окрестности для получения выходного значения в текущей точке [2].

Фильтры позволяют накладывать на изображения различные эффекты: размытие, подавления шума, деформацию и др.

Рассмотрим подробнее эффект повышения резкости изображения. Цель повышения резкости изображения заключается в том, чтобы подчеркнуть яркостные переходы. Пусть $f:\mathbb{Z}\to\mathbb{Z}$ – некоторая дискретная функция. В [2] показано, что для выделения переходов можно использовать производные дискретных функций следующего вида:

$$\frac{df}{dx} = f(x+1) - f(x),$$

$$\frac{d^2 f}{dx^2} = f(x+1) + f(x-1) - 2f(x). \qquad (1)$$

а простейшим оператором, который можно использовать для осуществления повышения резкости, – оператор Лапласа

$$\nabla^2 f = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2} \qquad . (2)$$

Первая производная f'_x удовлетворяет следующим свойствам: равна нулю в областях с постоянным уровнем яркости, отлична от нуля в начале, середине и конце области яркости.

Для второй производной f''_{xx} имеют место свойства: равна нулю на участках, где нет изменения яркости; отлична от нуля в начале и конце области изменения яркости и равна нулю в областях изменения яркости на постоянную величину. Таким образом, первая производная показывает скорость изменения яркости изображения, а вторая – места начала и конца изменения яркости. Зная эти места, можно «усилить» различие яркости, что и дает повышение резкости.

Для описания границ изменения яркости используется маска, которая является результатом применения оператора Лапласа к изображению [2]. Пусть $f(x,y)$ – исходное изображение, $g(x,y)$ – обработанное изображение. Повышение резкости с помощью лапласиана осуществляется по формуле

$$g(x,y) = f(x,y) - \nabla^2 f(x,y).$$

Подставляя в (2) вторые производные с учетом того, что изображение – это функция двух переменных, получаем дискретный вид лапласиана двух переменных

$$\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y). \qquad (3)$$

Формуле (3) соответствует следующее матричное представление:

$$\nabla^2 f = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Для включения диагональных элементов в формулу (3) добавляются еще два члена вида (1):

$$\frac{d^2 f}{dx^2} = f(x-1, y-1) + f(x+1, y+1) - 2f(x, y),$$
$$\frac{d^2 f}{dy^2} = f(x+1, y-1) + f(x-1, y+1) - 2f(x, y).$$

В итоге получим матричное представление лапласиана следующего вида:

$$\nabla^2 f = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \quad (4)$$

Повышение резкости изображения осуществляется для каждого пикселя изображения на основе следующей процедуры:

1. Выбирается пиксель $x_{ij} = \left\langle x_{ij}^R, x_{ij}^G, x_{ij}^B \right\rangle$ изображения $X_{n\times n}$, окрестность $P^{x_{ij}}$ которого имеет вид:

$$P^{x_{ij}} = \begin{pmatrix} x_{i-1,j-1} & x_{i-1,j} & x_{i-1,j+1} \\ x_{i,j-1} & x_{i,j} & x_{i,j+1} \\ x_{i+1,j-1} & x_{i+1,j} & x_{i+1,j+1} \end{pmatrix}.$$

2. Компоненты каждого элемента матрицы $P^{x_{ij}}$ умножаются на соответствующие им коэффициенты лапласиана (4). После этого элементы результирующей матрицы покомпонентно складываются; полученный элемент является новым значения яркости пикселя x_{ij}:

$$x_{ij}^{R\prime} = \sum_{i=1}^{3}\sum_{j=1}^{3} \nabla^2 f_{ij} \cdot x_{i-2,j-2}^R,$$
$$x_{ij}^{G\prime} = \sum_{i=1}^{3}\sum_{j=1}^{3} \nabla^2 f_{ij} \cdot x_{i-2,j-2}^G,$$
$$x_{ij}^{B\prime} = \sum_{i=1}^{3}\sum_{j=1}^{3} \nabla^2 f_{ij} \cdot x_{i-2,j-2}^B.$$

После вычисления значений яркостей каждого цвета необходимо проверить, что они не вышли за пределы допустимых значений:

$$x_{ij}^{R\prime\prime} = \min\left(255, \max\left(0, x_{ij}^{R\prime}\right)\right),$$
$$x_{ij}^{G\prime\prime} = \min\left(255, \max\left(0, x_{ij}^{G\prime}\right)\right),$$
$$x_{ij}^{B\prime\prime} = \min\left(255, \max\left(0, x_{ij}^{B\prime}\right)\right).$$

В случае, если пиксель x_{ij} находится на нет некоторых соседних пикселей (при $i, j = 0, n$), значения их яркостей полагаются равными 0. Результатом является изображение, у которого изменены пиксели на границах изменения яркостей.

1.2. Нейросетевая обработка изображений

Наибольшую популярность в задачах сегментирования медицинских изображений обрела нейронная сеть U-Net, созданная в 2015 во Фрайбургском университете Германии специально для задач анализа медицинских изображений [5]. Особенностью данной сети является нали-

чие свёрточных слоев и слоев субдискретизации. Выходом каждого слоя является *карта признаков* – матриц, описывающих наличие на изображении тех или иных деталей. В глубоких (более одного скрытого слоя) сетях происходит переход от конкретных деталей на изображении к более абстрактным.

Слой свёртки включает в себя фильтр, называемый *ядрами свёртки*. Ядро представляет собой матрицу размера $L \times L$, где L – нечетное положительное число, и предназначено для определения наличия на изображении определенного признака и места его расположения. Коэффициенты этой матрицы – параметры нейронной сети, которые определяются во время обучения. Слой может включать несколько ядер, что позволяет одновременно определять наличие и положение нескольких признаков. Ядро K обрабатывает вход X размера $n \times n, n \in \mathbb{N}$ предыдущего слоя путем прямого прохода по матрице признаков и вычисления суммы поэлементного умножения элементов фильтра и подматрицы ядра по формуле

$$y_{ij} = \sum_{p=1}^{L} \sum_{q=1}^{L} (X_{ij}^{L} \circ K)_{pq},$$

где $X_{ij}^{L} = \begin{pmatrix} x_{i-L,j-L} & \cdots & x_{i-L,j-1} & x_{i-L,j} & x_{i-L,j+1} & \cdots & x_{i-L,j+L} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \iddots & \vdots \\ x_{i-1,j-L} & \cdots & x_{i-1,j-1} & x_{i-1,j} & x_{i-1,j+1} & \cdots & x_{i-1,j+L} \\ x_{i,j-L} & \cdots & x_{i,j-1} & x_{i,j} & x_{i,j+1} & \cdots & x_{i,j+L} \\ x_{i+1,j-L} & \cdots & x_{i+1,j-1} & x_{i+1,j} & x_{i+1,j+1} & \cdots & x_{i+1,j+L} \\ \vdots & \iddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{i+L,j-L} & \cdots & x_{i+L,j-1} & x_{i+L,j} & x_{i+L,j+1} & \cdots & x_{i+L,j+L} \end{pmatrix}$,

$i, j = L, \ldots, n - L$ – элементы, обрабатываемые фильтром;

y_{ij} – элемент выходной матрицы Y размера $(n-L) \times (n-L)$.

Выходом свёрточного слоя является набор карт признаков, каждая из которых описывает наличие и положение деталей на изображении, которые могут опознать ядра свёртки.

Слой субдискретизации представляет собой уплотнение карты признаков, при этом группа пикселей уплотняется до одного пикселя на основе нелинейного преобразования. Если X – карта признаков размера $n \times n$, то элемент y_{ij} выхода слоя Y' размера $\frac{n}{2} \times \frac{n}{2}$ определяется по формуле

$$y_{ij} = \max\left\{x_{2i,2j}, x_{2i+1,2j}, x_{2i,2j+1}, x_{2i+1,2j+1}\right\}.$$

Отличительной особенностью сети U-Net заключается в наличии сжимающего (слева) и расширяющего (справа) частей (рис. 1). Количество карт признаков на каждом шаге удваивается. Сжимающий путь представляет собой типовую свёрточную сеть. Он содержит два подряд свёрточных слоя размерности 3×3, после которых идут слои субдискретизации. Каждый шаг расширяющего пути содержит слой, обратный слою субдискретизации, который расширяет карту признаков. После этого слоя следует свёрточный слой размерности 3×3, уменьшающий количество каналов признаков. Далее идет конкатенация с соответствующим образом обрезанной картой признаков из сжимающего пути и два свёрточных слоя размерности 3×3. Всего эта сеть имеет 23 слоя.

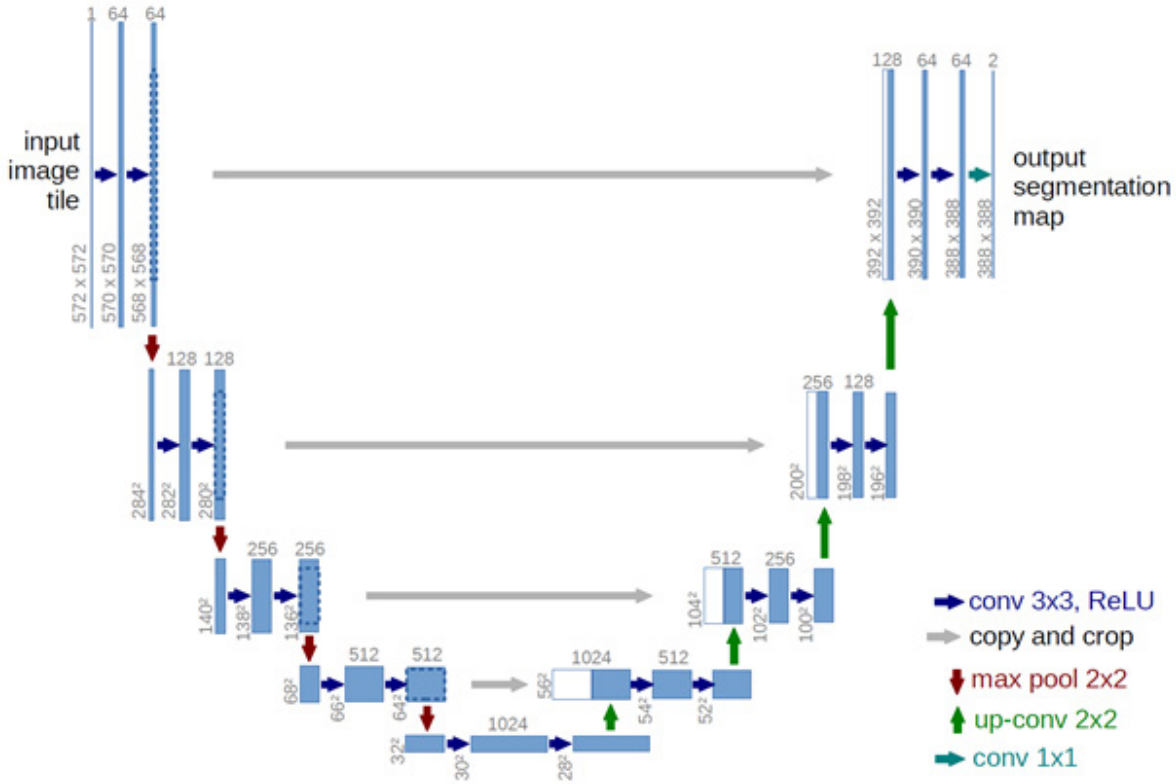


Рис. 1. Архитектура сети U-Net [5]

2. Результаты и их обсуждение

Для проведения исследования рассматриваются изображения из набора Data Science Bowl 2018 [8]. На них представлены клетки живых организмов с четко выраженными ядрами. К каждому изображению также идет набор масок, показывающий местоположение каждого ядра. Данный набор изображений требует предварительной обработки. Поскольку на изображении может быть несколько ядер и для каждого из них существует отдельная маска, все маски объединяются в одну путем наложения их друг на друга, что позволяет находить все ядра клеток сразу.

Для реализации нейронной сети было использовано следующее программное и аппаратное обеспечение:

1) язык программирования **Python 3**, как наиболее часто используемый при решении задач машинного обучения;

2) библиотека **Keras**, позволяющая реализовать различные модели нейронных сетей с помощью комбинирования готовых реализаций слоёв и использования готовых реализаций алгоритмов оптимизации;

3) видеокарта nVidia GeForce GT930MX, поддерживающая технологию **nVidia CUDA**, которая позволяет производить вычисления на графических ядрах и, тем самым, многократно повысить скорость обучения нейронных сетей.

Для обучения нейронной сети были выбраны следующие параметры:

а) функция потерь – бинарная кросс-энтропия, показывающая, насколько класс пикселя, определенный нейронной сетью, отличается от истинного класса, и вычисляемая по формуле

$$E = -\frac{1}{2}\left(y \log \hat{y} + (1-y)\log(1-\hat{y})\right),$$

где y – истинный класс пикселя,

$\hat{y}$ – предсказанный класс пикселя [5];

b) метод оптимизации – Adam, являющийся модификацией стохастического градиентного метода [5];

c) метрика для оценки точности предсказания – Intersection over Union (IoU), являющаяся отношением площади пересечения ожидаемой и предсказанной маски объектов к площади объединения этих масок:

$$I = \frac{\left|Y \cap \hat{Y}\right|}{\left|Y \cup \hat{Y}\right|},$$

где Y – ожидаемая маска объекта,

$\hat{Y}$ – предсказанная нейронной сетью [9].

Набор данных, состоящий из 670 изображений клеток и масок изображений, был разделен случайным образом на обучающую и тестовую выборку, включающие 80 % и 20 % изображений соответственно. Процесс обучения прекращается по истечении 10 эпох обучения.

На графике (рис. 2) приведены значения точности после каждой эпохи обучения для двух нейронных сетей, одна из которых обучалась на исходных изображениях, а другая – на предобработанных.

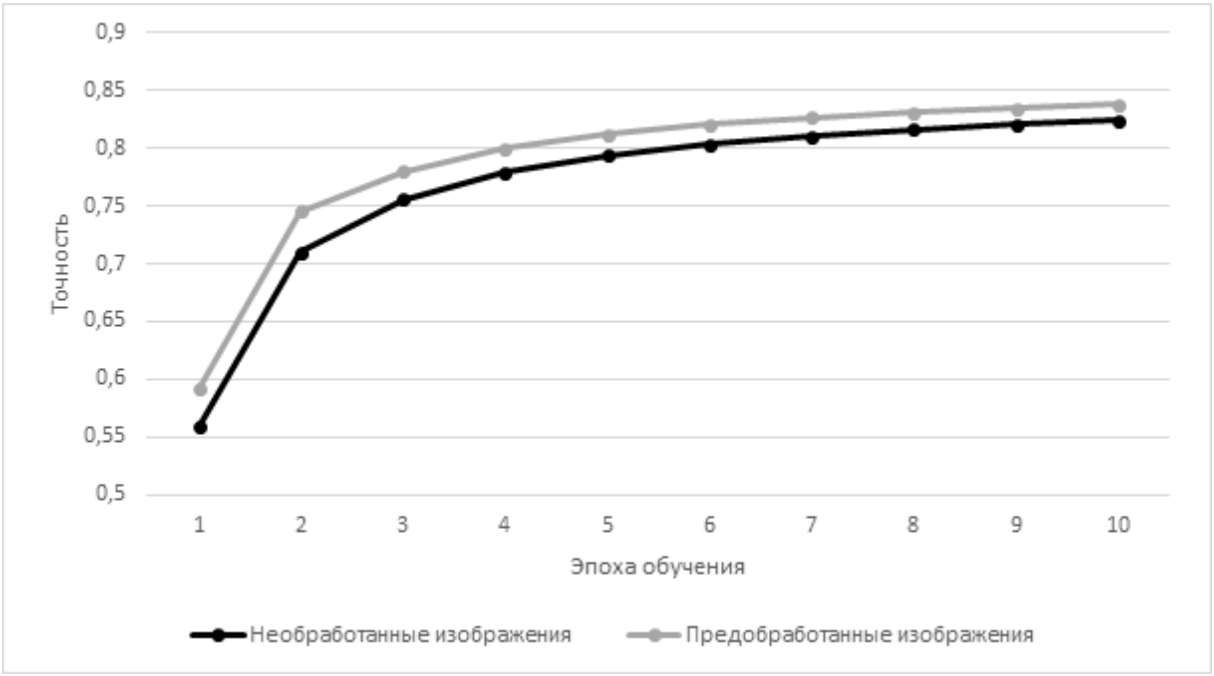


Рис. 2. Точность предсказания после каждой эпохи обучения

Рассмотрим пример. На рис. 3 представлено исходное и предобработанное с повышенной резкостью изображения. На рис. 4 представлена исходная маска изображения и маска, полученная от нейронной сети, обученной на исходных изображениях, а на рис. 5 – от нейронной сети, обученной на предобработанных.

Сравнение сгенерированных масок показывает, что нейронная сеть, обученная на предобработанных изображениях, дает более четкие контуры ядер клеток, нежели нейронная сеть, обученная на исходных изображениях. Таким образом, повышение резкости изображений позволяет получать более точную сегментацию изображений в течение одного и того же количества эпох обучения.

Также результаты эксперимента показывают, что на предобработанных изображениях получить требуемую точность сегментации можно за меньшее количество эпох обучения, чем

Рис. 3. Исходное и предобработанное изображения

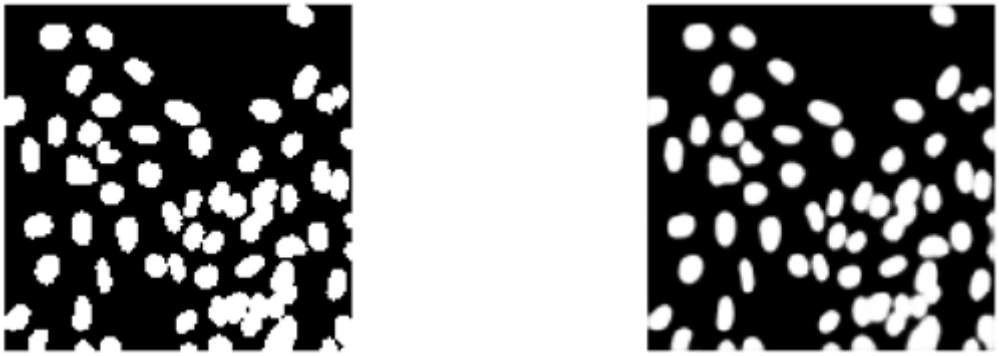


Рис. 4. Ожидаемая маска изображения и сгенерированная обученной на исходных изображениях нейронной сетью

Рис. 5. Ожидаемая маска изображения и сгенерированная обученной на предобработанных изображениях нейронной сетью

на исходных изображениях. Например, значение точности 0,8 при обучении на исходных изображениях достигнуто за 6 эпох, а на предобработанных – за 4. Это позволяет сократить время на обучение нейронных сетей, если задана необходимая точность предсказаний.

Заключение

В статье рассмотрено влияние предобработки изображений с помощью цифровых фильтров на точность их сегментации с помощью свёрточных нейронных сетей. Повышение резкости рассмотренных медицинских изображений позволило повысить точность предсказания нейронной сети маски уже после первых итераций обучения в среднем на 2 %. В будущем планируется исследовать возможность предобработки с помощью цифровых фильтров изображений, а также с использованием нейронных сетей типа автокодировщиков, с объектами иного рода (спутниковые снимки, дорожная обстановка и т. д.).

Литература

1. *Shapiro, L.* Computer Vision. / L. Shapiro, G. Stockman. – Нью-Джерси, 2001. – 580 с.

2. *Гонсалес, Р.* Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – 3-е изд., испр. и доп. – М. : Техносфера, 2012. – 1104 с.

3. *Felzenszwalb, P.* Efficient Graph-Based Image Segmentation // P. Felzenszwalb, D. Huttenlocher // International Journal of Computer Vision. – 2004.

4. *Barghout, L.* Real-world scene perception and perceptual organization: Lessons from Computer Vision // L. Barghout, J. Sheynin // Journal of Vision. – 2013. – Vol. 13.

5. *Гудфеллоу, Я.* Глубокое обучение / Я. Гудфеллоу, Бенджио И., Курвилль А. – 2-е изд., испр. – М. : ДМК Пресс, 2018. – 652 с.

6. U-Net: Convolutional Networks for Biomedical Image Segmentation: [сайт]. – (URL: https://arxiv.org/abs/1505.04597)

7. *Гонсалес, Р.* Цифровая обработка изображений в среде MATLAB / Р. Гонсалес, Р. Вудс, С. Эддинс. – М. : Техносфера, 2006. – 616 с.

8. 2018 Data Science Bowl: [сайт]. – (URL: https://www.kaggle.com/c/data-science-bowl-2018) (дата обращения: 28.02.2020).

9. *Rezatofighi, H.* Generalized Intersection over Union: A Metric and A Loss for Bounding Box // H. Rezatofighi, T. Nathan, JunYoung G. etc. // The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2019.

Амелин Игорь Евгеньевич – студент 2-го курса магистратуры кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: i.amelin95@yandex.ru

Леденёва Татьяна Михайловна (научный руководитель) – д-р техн. наук, проф., профессор кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: tm-ledeneva@yandex.ru

УДК 004.032

РАЗРАБОТКА СТРУКТУРЫ ИНФОРМАЦИОННОЙ СИСТЕМЫ «ДОКУМЕНТЫ СТУДЕНТА» ДЛЯ ВОРОНЕЖСКОГО ГОСУДАРСТВЕННОГО ТЕХНИЧЕСКОГО УНИВЕРСИТЕТА

К. А. Андреева, Н. В. Акамсина

Воронежский государственный технический университет

Введение

Сфера информационных технологий с каждым днем все больше и больше внедряется в жизнь человека. Сегодня нельзя представить ни одну отрасль человеческой деятельности без применения автоматизации ручного труда человека, это коснулось и учебных заведений. За все время обучения, студента сопровождает большое количество документации – студенческий билет, аттестационные ведомости, зачетная книжка, научные публикации, грамоты участия в конференциях, хакатонах, гранты и многие другие. Выбранная тема является актуальной в современном информационном обществе, поскольку нельзя представить ни одного студента, который не хотел бы иметь электронный пропуск, зачетную книжку, читательский билет в одном централизованном месте хранения.

1. Описание разработки информационной системы

Для начала, необходимо разобраться с понятием информационная система. Обратимся к учебнику Избачкова Ю. С. «Информационные системы», в соответствии с ним: информационная система – прикладная программная система, ориентированная на сбор, хранение, поиск и обработку текстовой и/или фактографической информации [1]. Чудинов И. Л. дает свое определение – информационная система – совокупность технического, программного и организационного обеспечения, а также персонала, предназначенного для того, чтобы своевременно обеспечить надлежащих людей надлежащей информацией [2]. В соответствии с приведенными определениями, можно сделать вывод о том, что информационная система – совокупность, средств и методов, предназначенных для хранения, поиска и обработки информации для достижения поставленной цели.

Информационная система «Документы студента» представляет из себя платформу, включающую микро сервисы, взаимодействующие с системой через API. Прежде чем перейти к проектированию информационной системы «Документы студента», необходимо провести анализ предметной области. На сервисе Play Market существует множество подобных приложений. Был проанализирован ряд подобных программ других ВУЗов, а именно: Студент СФУ, МГИМО, КОНФРЯЗГМУ, МТИ – в них не реализован модуль с хранением документов, в основном все эти приложения имеют только модуль с расписанием, новостями ВУЗа и учебными планами. На рис. 1 приведены основные характеристики приложений, в которых реализован модуль для хранения документов.

Проанализированные приложения, в основном включают в себя функцию просмотра и добавления личных достижений. Реализованные функции:

- Зачетная книжка: хранит информацию о дисциплине, дате сдачи, семестре, виде контроля, оценки.
- Достижения студента: включают в себя грамоты, научные статьи.

Характеристики	Белгородский ГАУ	Университет ИТМО	М портал	ТГУ им. Державина
Электронный пропуск	"-"	"-"	"-"	"-"
Просмотр оценок по дисциплинам	"+"	"+"	"+"	"+"
Просмотр и добавление личных достижений	"+"	"-"	"-"	"+"
Просмотр и добавление научных работ	"+"	"-"	"-"	"+"

Рис. 1. Основные характеристики приложений

При проектировании любой системы необходимо учитывать ее дальнейшее развитие, поэтому она должна быть гибкой и легко изменяемой [4]. Общая структура проектируемого приложения, представлена на рис. 2.

Рис. 2. Общая структура приложения

Разрабатываемое приложение поможет облегчить жизнь студента, рассмотрим каждый модуль в отдельности. Модуль «Мои достижения» обеспечивает централизованное хранение личных достижений, полученных в течении всего времени обучения, к ним могут относиться: научно-исследовательские статьи, грамоты за участие в конкурсах, грантах, хакатонах, патенты и т. д. Данный раздел представляет из себя некое электронное портфолио. Электронное портфолио – это популярный современный способ интеграции, визуализации и оценки как профессионально-образовательных достижений индивида, так и его общественных навыков [6].

Модуль «Электронный пропуск» поможет студенту проходить через турникеты без пластиковых карт, а использовать свой телефон или смарт-часы, с поддержкой системы NFS. Данная технология получила широкое распространение в странах Запада. Для работы такого модуля необходимо специально оборудованные турникеты [5].

Модуль «Зачетная книжка» предназначен для просмотра успеваемости по дисциплинам, подразумевается интеграция данного модуля с программой 1С: Университет.

Модуль «Библиотека» предоставляет возможность студенту просматривать свои задолженности по книгам, вход в личный кабинет осуществляется по электронному пропуску.

Для работы системы необходимо взаимодействие со сторонними системами через API, например: 1С:Университет, база данных электронных пропусков, информационная система библиотеки, для предоставления сведений о студенте, таких как: фамилия, имя, отчество, дата рождения, дата поступления в ВУЗ, номер приказа, группа, курс, сведения об успеваемости, индивидуальные достижения, взятие книги в библиотеке. Интерфейсом для взаимодействия программы будет служить API- это граничный слой, чье прикладное программное обеспечение использует средства языков программирования для вызова сервисов [3]. На рис. 3 приведена схема взаимодействия API приложения (на примере программного обеспечения ВГТУ).

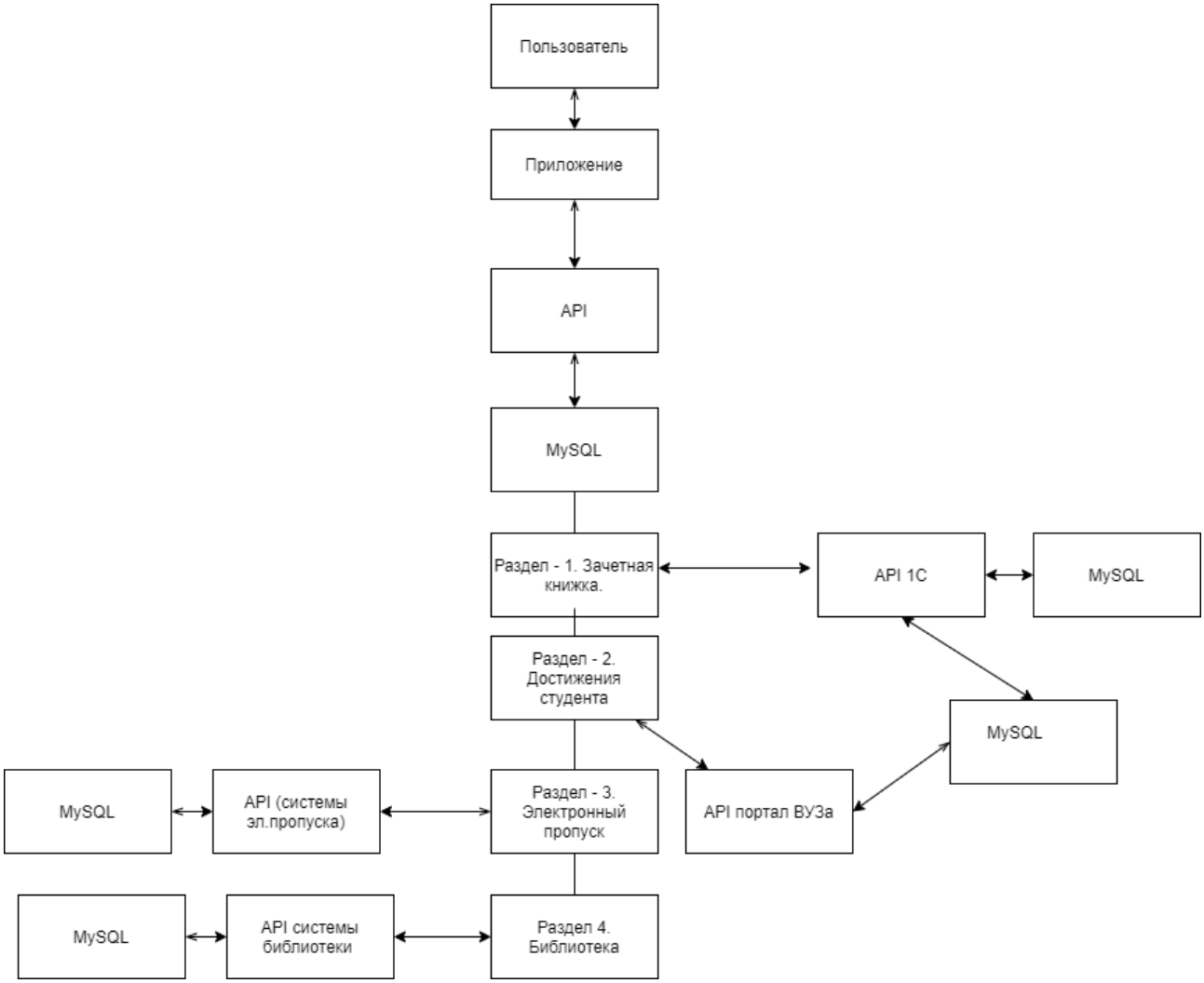


Рис. 3. Схема взаимодействия компонентов API

Заключение

Таким образом, разрабатываемое приложение поможет студентам хранить необходимые документы централизовано в одном месте. Для реализации такой информационной системы необходимо взаимодействие с внутренними программами ВУЗа, поскольку система «Мои документы» является платформой, предоставляющей доступ ко всем необходимым данным.

Литература

1. *Избачков, Ю. С.* Информационные системы для вузов: учебное пособие / Ю. С. Избачков, В. Н. Петров. – СПб. Изд-во Питер, 2005. – 656 с.

2. *Чудинов, И. Л.* Информационные системы и технологии: учебное пособие / И. Л. Чудинов, В. В. Осипова; Томский политехнический университет. – Томск : Изд-во Томского политехнического университета, 2013. – 145 с.

3. Информационные технологии (ИТ). Системы позиционирования в реальном времени (RTLS). Часть 1. Прикладной программный интерфейс (API). ГОСТ Р ИСО/МЭК 24730-1-2017. – Введ.2018-12-01, переиздан – январь 2019. – Москва : Изд-во стандартов, 2018. – 40 с.

3. *Колыхалова, Е. В.* Анализ жизненного цикла продукта с учетом будущих изменений // Колыхалова Е. В. Информационные технологии в строительных, социальных и экономических системах: научно-технологический журнал / Е. В. Колыхалова, К. А. Андреева. – Воронеж, 2019 – С. 57–59.

4. *Гордеев, С.* NFS университеты будущего. – Режим доступа: http://secuteck.ru/articles2/sys_ogr_dost/nfc-universitety-buduschego – свободный (дата обращения 13.03.2020).

5. *Круглякова, А. А.* Технология визуализации и оценки личных достижений студента / А. А. Круглякова О. В.Курипта // Информационные технологии в строительных, социальных и экономических системах: научно-технический журнал. – 2018. – С. 117–121.

Андреева Кристина Алексеевна – студентка 3 курса кафедры систем управления и информационных технологий в строительстве Воронежского государственного технического университета. E-mail: kandreeva952@gmail.com

Акамсина Надежда Валериевна – канд. техн. наук, доцент кафедры систем управления и информационных технологий в строительстве Воронежского государственного технического университета. E-mail: akamsina@vgasu.vrn.ru

УДК 004.942

АРХИТЕКТУРА ПРЕДПРИЯТИЯ, КАК ИНСТРУМЕНТ ПОСТРОЕНИЯ СТРУКТУРНО-ИНФОРМАЦИОННОЙ МОДЕЛИ ИСПОЛНИТЕЛЬНОГО ОРГАНА ГОСУДАРСТВЕННОЙ ВЛАСТИ

Н. Г. Аснина, А. В. Хрюкина

Воронежский государственный университет

Введение

На сегодняшний день термин «Электронное правительство» используется для обозначения всего, от «онлайн-государственных услуг» до «обмена информацией и услугами в электронном виде с гражданами, предприятиями и другими органами власти». Традиционно электронное правительство рассматривается как использование ИКТ для повышения эффективности деятельности государственных органов и предоставления государственных услуг в режиме онлайн, для проведения широкого круга взаимодействий с гражданами и предприятиями, а также открытые правительственные данные и использование ИКТ для обеспечения инноваций в управлении.

Информационные и коммуникационные технологии стали частью современных управленческих систем во всех сферах государственного управления. Главной проблемой является необходимость сокращения дублирования информационных потоков при взаимодействии органов власти. Поэтому поставлена задача на создание и развитие информационной системы для сбора и анализа данных, обмена ими.

1. Архитектурный подход к построению структурно-информационной модели ИОГВ

Современные компьютерные и технологические системы сложны, особенно когда они отвечают за большую часть производственной деятельности в большинстве организаций. Физические и облачные серверы, приложения и корпоративное программное обеспечение должны беспрепятственно обмениваться данными, чтобы предоставить клиенту успешный и удовлетворительный пользовательский опыт.

Целостное представление всех этих частей называется архитектурой предприятия. Рекомендации по этой реализации часто называют моделями архитектуры предприятия.

Проще говоря, модели архитектуры предприятия относятся к любой структуре, процессу или методологии, которая информирует о том, как создавать и использовать корпоративную архитектуру.

На высоком уровне архитектура предприятия предлагает комплексный подход и целостное представление об информационных технологиях в масштабах всего предприятия. Предприятие – это бизнес, компания, фирма или группа любого размера, которые предоставляют потребителям товары и/или услуги. Это может также включать любое организованное подразделение, имеющее общую цель, например, отраслевой консорциум или некоммерческая группа.

Во многих средах, которые служат для создания корпоративной архитектуры, цель комплексного подхода всегда заключается в успешном выполнении стратегии с производительностью, эффективностью, безопасностью, долговечностью и гибкостью.

Архитектура предприятия состоит из бизнес-архитектуры, архитектуры информации, архитектуры приложений и технологической архитектуры (рис. 1) [1].

Рис. 1. Структура архитектуры предприятия

Бизнес-архитектура является составляющей частью АП и отвечает за стратегии, цели и миссии. Она неразрывно связана с процессом его управления. Под управлением предприятием обычно понимается деятельность, которая регулируется нормативными документами. Соответственно, предприятие не может существовать без организационной структуры. Бизнес-архитектура описывает деятельность предприятия с точки зрения ключевых бизнес-процессов.

Архитектура информации характеризуется набором методик и инструментов, который описывает информационную модель предприятия. Она включает базы данных и хранилища данных, а также информационные потоки (как внутри организации, так и связи с внешним миром).

Архитектура приложений определяет, какие приложения используются и должны использоваться для управления данными.

Технологическая архитектура формируется из совокупности программно-аппаратных средств, методов и стандартов, обеспечивающих эффективное функционирование приложений информационной системы.

2. Структурно-информационная модель ИОГВ (шаблон)

Основополагающий принцип электронного правительства, подкрепляемый эффективной институциональной основой электронного управления, заключается в улучшении внутренней работы государственного сектора путем сокращения финансовых затрат и времени операций, с тем чтобы лучше интегрировать рабочие потоки и процессы и обеспечить эффективное использование ресурсов различными учреждениями государственного сектора в целях принятия устойчивых решений [2].

При этом, главной проблемой является необходимость сокращения дублирования информационных потоков при взаимодействии органов власти. Поэтому цель работы является весьма актуальной.

На рис. 2 представлена информационная модель исполнительного органа государственной власти, которая строилась на основе архитектурного подхода к исследованию предметной области.

Архитектурный подход предполагает выделение 4х аспектов, характеризующих специфику изучаемой области. В нашем случае 1 аспект (бизнес-архитектура) представлен набором: наименование, нормативные правовые акты, организационная структура и виды деятельности. Бизнес-архитектура включает в себя архитектуру бизнес-процессов, которая представлена

Рис. 2. Информационная модель исполнительного органа государственной власти

видами деятельности. На основании задач, функций и услуг появляются взаимодействия с исполнительными органами государственной власти, которые отражают архитектуру информации. Пункт 6 включает в себя последний аспект, а именно архитектуру приложений.

Данная модель призвана упорядочить процедуру выявления «легальных» и «легитимных» информационных потоков создаваемых и потребляемых ИОГВ.

В рамках исследования на основе предложенного выше шаблона была разработана структурно-информационная модель департамента социальной защиты Воронежской области.

Дополнительным источником информации для проведения этой работы послужила анкета, разработанная сотрудниками департамента цифрового развития ВО, которая представлена в табл. 1.

3. Концепция технологической платформы государственного управления

Предполагается, что технологическая платформа государственного управления будет выступать в качестве интеграционной шины, которая упростит слияние государственных информационных систем и баз данных.

На данный момент не каждый исполнительный орган государственной власти обеспечен информационными системами, поэтому документооборот реализован только в бумажном виде. В этом случае появляются два варианта подключения к интеграционной шине. Первый

Таблица 1

Анкета

Вопрос	**Пояснение вопроса**
Наименование функциональной задачи (полномочия)	Укажите наименование функциональной задачи (полномочия)
Описание функциональной задачи (полномочия)	Опишите функциональную задачу (полномочие), дайте пояснения
Полномочие организации для реализации функциональной задачи	Укажите пункт положения об организации, на основании которого выполняется функциональная задача
Нормативно-правовое обоснование	Укажите пункт и реквизиты НПА, на основании которого реализуются полномочия по выполнению функциональной задачи
Наименование автоматизированных систем формирования отчета	Укажите наименование информационных систем, которые были использованы при выполнении функциональной задачи. Если информационные системы не использовались, поле не заполняется
Используемые справочники	Укажите тип используемых справочников (общероссийские, региональные, ведомственные) и их наименование.
Потребитель информации	Укажите наименование организации, которой представляются результаты выполнения функциональной задачи
Способ направления информации потребителю (бумажное письмо, электронный документооборот, электронная почта, информационная система)	Укажите способ направления результатов выполнения функциональной задачи
Периодичность направления информации потребителю	Укажите требуемую периодичность представления информации потребителю: ежедневно, ежемесячно, ежеквартально, ежегодно
Поставщик сведений для отчета (наименование организаций)	Укажите наименование организации, которая представила информацию для выполнения функциональной задачи. Если дополнительная информация не запрашивалась, поле не заполняется
Способ получения сведений от поставщика (бумажное письмо, электронный документооборот, электронная почта, информационная система)	Укажите способ направления информации для выполнения функциональной задачи. Поле заполняется, если запрашивалась дополнительная информация для выполнения функциональной задачи
Структурное подразделение, ответственное за выполнение задачи	Укажите наименование структурного подразделения, ответственного за выполнение задачи

вариант включает в себя информационную систему, в которой есть возможность выгрузки данных в общую стандартизированную систему. Во втором варианте непосредственно представлен прямой доступ к этой системе, в рамках которого происходит ввод данных.

Технологическая платформа государственного управления должна собирать, обрабатывать и хранить данные, на основании запросов формировать документы, проводить анализ для выявления статистических данных.

Концепция технологической платформы государственного управления представлена на рис. 3.

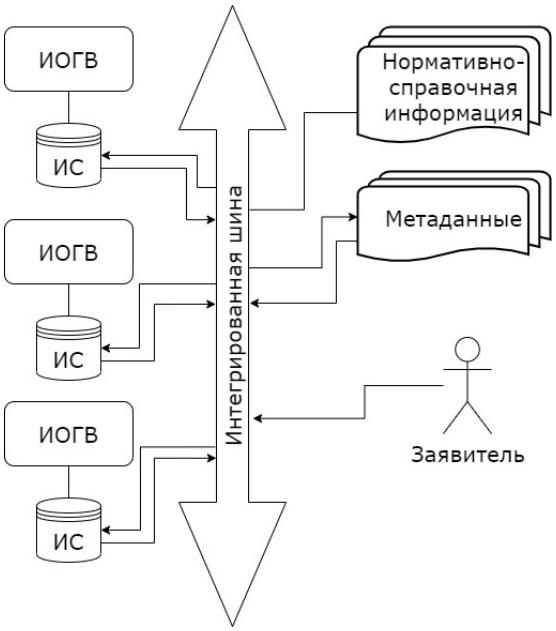


Рис. 3. Концепция технологической платформы государственного управления

Регистр данных будет являться ключевым звеном, который объединяет в себе ссылки на данные из информационных систем, баз данных, справочников и запросов. Концептуальная модель технологической платформы государственного управления в форме ER-модели изображена на рис. 4 [2].

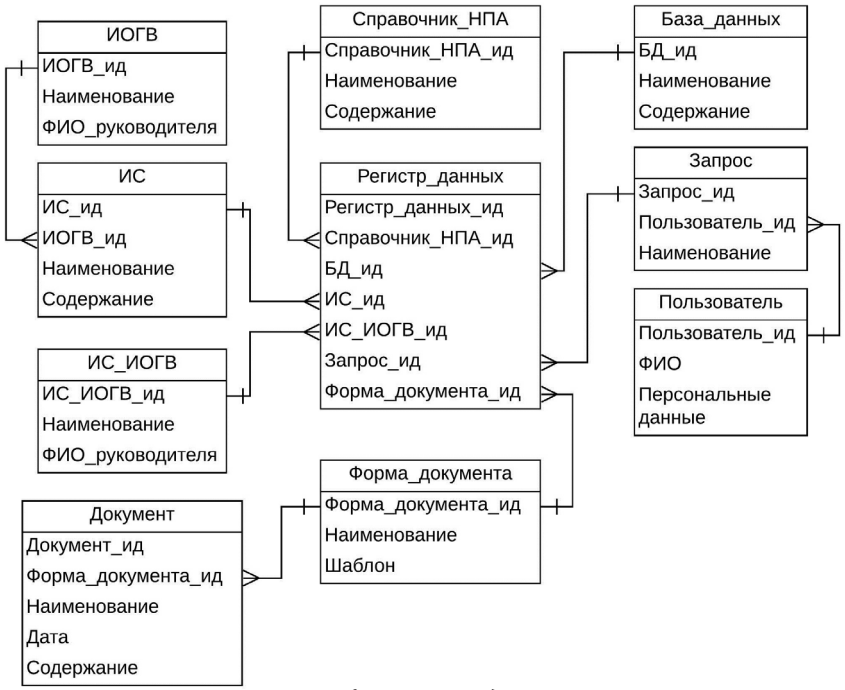


Рис. 4. Концепция технологической платформы государственного управления в виде ER-модели

Заключение

В данной работе наглядно продемонстрирована целесообразность применения архитектурного подхода к разработке структурно-информационных моделей различного вида бизнес-структур, в том числе и некоммерческих.

Литература

1. *Зараменских, Е. П.* Архитектура предприятия : учебник для бакалавриата и магистратуры / Е. П. Зараменских, Д. В. Кудрявцев, М. Ю. Арзуманян ; под редакцией Е. П. Зараменских. – Москва : Издательство Юрайт, 2018. – 410 с. – Текст : электронный // ЭБС Юрайт [сайт]. – URL: https://urait.ru/bcode/412299 (дата обращения: 20.04.2020).

2. *Ларман, Крэг.* Применение UML и шаблонов проектирования : Введение в объектно-ориентированный анализ и проектирование : [Пер. с англ.] / Крэг Ларман [и др.]. – М. : Вильямс, 2001. – 489 с.

3. Об оптимизации структуры исполнительных органов государственной власти Воронежской области: Указ губернатора Воронежской области от 12 декабря 2018 № 733-у. – URL: http://pravo.govvrn.ru/content указ-губернатора-воронежской-области-от-12122018-№-733-у.

Хрюкина Анна Владимировна – магистрант 1-го года обучения кафедры математических методов исследования операций Воронежского государственного университета. E-mail: khryukina.anna@gmail.com

Аснина Наталия Георгиевна (научный руководитель) – канд. техн. наук, доц., доцент кафедры математических методов исследования операций Воронежского государственного университета. E-mail: mmio@amm.vsu.ru

УДК 77

РАСКРАШИВАНИЕ ЧЕРНО-БЕЛЫХ ИЗОБРАЖЕНИЙ

О. А. Безрукова

Воронежский государственный университет

Введение

В современном мире для преобразования изображений из черно-белых в цветные зачастую используют специализированные программы, такие как Photoshop, GIMP, Movavi Photo Editor и другие. Однако, все действия приходится делать вручную, что значительно усложняет процесс обработки фотографии. Для того, чтобы раскрасить одну фотографию потребуются знания какой-либо программы, умение подбирать и сочетать цвета, а также время. Отсюда можно сделать вывод о том, что необходимо создать такое приложение (или онлайн сервис), который мог бы за считанные секунды преобразовать черно-белое изображение в цветное с достаточно высокой точностью.

1. Выбор архитектуры

Перед тем как приступить к реализации приложения, необходимо изучить существующие методы и подходы к созданию подобных программ. Все похожие продукты основаны на технологии машинного обучения, но используют различные архитектуры. К самым распространенным архитектурам (на основе различных исследовательских работ и приложений с открытым исходных кодом) можно отнести следующие:

1) на изображение вручную добавляются маленькие точки, которые являются подсказкой для нейронной сети;

2) поиск похожих изображений и перенос цветов с найденного изображения на пользовательское;

3) объединение итоговой классификации между кодировщиком и декодировщиком.

Путем исследований была выбрана архитектура со слоем слияния (третий пункт в приведенном выше списке), так как давала лучшие результаты и более понятна в реализации.

2. Предварительные опыты

Для работы с изображениями необходимо было создать несколько тестовых программ, чтобы лучше понять как преобразуются цвета и как научить нейронную сеть различать оттенки.

2.1. Обработка одной фотографии

На вход программе подается картинка в цветовом пространстве RGB, которая при выполнении программы преобразуется в Lab, для удобства работы с ней. Изображение в цветовом пространстве Lab содержит L слой – черно-белое изображение, которое отвечает за яркость, а цветные слои RGB объединяются в два – от зеленого до красного и от синего до желтого.

Далее программа использует L слой как черно-белое изображение и с помощью сверточных фильтров генерирует цветовые слои a и b. После того, как сгенерировались цветные слои, они

сравниваются с теми слоями, которые получились при преобразовании исходного изображения, то есть с точными значениями. Таким образом получается погрешность, которую нейронная сеть с каждой эпохой обучения пытается минимизировать. В итоге, после даже тридцати повторов выдается неплохой результат.

Однако, такая программа была необходима только для того, чтобы понять как преобразуются цвета и каким образом можно сопоставить черно-белое изображение и цветное. Она может раскрасить только ту картинку, на которой она обучалась, а при попытке подать ей любое другое изображение, выдавала неверно подобранные цвета.

2.2. Деление фотографии на оттенки

Следующим шагом необходимо обучить модель так, чтобы она могла обрабатывать любое изображение, поданное на вход пользователем. Для этого нужно научить ее выделять признаки и обобщать их. Исходное изображение разбивается на небольшие квадратики, затем ищутся похожие структуры и обобщаются. С помощью сверточной нейросети обработанные изображения комбинируются, для того, чтобы понять, что изображено на картинке.

На этапе обучения нейросеть отталкивается от самых плохих результатов, чтобы с каждым разом улучшать показатели. Сначала она назначает каждому пикселю любой цвет, а затем, исходя из ошибок, корректирует фильтры, чтобы повысить точность. Изначально изображение окрашивается в коричневый цвет, так как на нем получается минимальное число ошибок. Также, в отличии от классифицирующей нейронной сети, которой важна итоговая классификация, поэтому в ходе обучения постепенно уменьшается размер и качество картинки, данная не изменяет исходное изображение.

Данная модель позволяет раскрашивать изображения монохромно, в основном с использованием коричневого цвета и близких к нему оттенков. Поэтому на следующем этапе необходимо обучить нейронную сеть различать цвета.

2.3. Обучение модели различать цвета на черно-белых изображениях

Для реализации данного этапа программа из прошлого шага была разбита на кодировщик и декодировщик, между которым находится слой слияния. Входное изображение проходит через кодировщик и классификатор Inception ResNet v2, который необходим для того, чтобы нейронная сеть могла сопоставить изображение на картинке с тем, как были раскрашены подобные объекты.

После обучения нейронной сети на небольшом количестве различных изображений были получены не очень хорошие результаты. Зачастую лицо человека было голубым, а деревья фиолетовыми, однако если для обучения выбрать похожие изображения, то черно-белую картинку модель раскрасить довольно неплохо.

Заключение

Таким образом, для того, чтобы достичь хороших результатов прежде всего необходимо иметь достаточно большой набор изображений для обучения (тренировочных данных), желательно высокого качества. Написание программ, описанных в первых двух этапах, помогло лучше понять как именно работает нейросеть с изображениями, на каких данных результаты лучше и сколько раз необходимо обучиться нейросети, чтобы получить на выходе неплохие результаты.

Говоря об актуальности данной работы можно сказать, что данная программа могла бы помочь людям упростить такую трудозатратную задачу, как раскрашивание черно-белых изображений. С ее помощью любой пользователь смог бы делать фото цветными.

Литература

1. DeOldify: программа для раскрашивания чёрно-белых изображений (https://habr.com/ru/post/428818/) (2018)

2. Документация библиотеки Tensorflow (https://www.tensorflow.org/)

3. ЦВЕТОВЫЕ МОДЕЛИ CMYK, RGB, LAB, HSB (https://sveres.ru/articles/prepress/tsvetovye-modeli-cmyk-rgb-lab-hsb.html)

4. Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными. – Вильямс, 2017. – 480 с.

5. Colorization Using Optimization (https://www.cse.huji.ac.il/~yweiss/Colorization/)

6. Automatic Colorization (https://tinyclouds.org/colorize/)

7. Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification (http://iizuka.cs.tsukuba.ac.jp/projects/colorization/data/colorization_sig2016.pdf)

Безрукова Ольга Алексеевна – студент 3-го курса кафедры Математического обеспечения ЭВМ Воронежского государственного университета. E-mail: katlinc@mail.ru

Светлана Юрьевна Болотова (научный руководитель) – канд. физ.-мат. наук, доцент кафедры Математического обеспечения ЭВМ Воронежского государственного университета. E-mail: Bolotova.svetlana@gmail.com

УДК 316.774

ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ ДЕЯТЕЛЬНОСТЬ В ГОДЫ ВЕЛИКОЙ ОТЕЧЕСТВЕННОЙ ВОЙНЫ

Н. А. Белов, А. Г. Ильченко

Воронежский государственный университет

Введение

В работе проведён анализ характера и средств информационно-аналитической деятельности в условиях Великой Отечественной войны по материалам книги начальника штаба 23-й стрелковой дивизии генерал-лейтенанта С. А. Андрющенко «Начинали мы на Славутиче».

Информационно-аналитическая деятельность «*заключается в осуществлении таких основных процессов как сбор, аналитико-синтетическая переработка, хранение и поиск закрепленной в документах …информации. А также в предоставлении или распространении этой информации…*» [2].

Не секрет, что в разные времена эта деятельность выглядела по-разному. Возьмем для примера процесс распространения информации. В наши дни наиболее популярным средством распространения информации является сеть Интернет, а у египтян это был папирус. Но сама сущность этой деятельности, ее структура – вещи неизменные.

1. Сбор информации разведкой

Начнем рассмотрение с такой составляющей этого процесса как сбор информации. Существуют различные его средства и способы, начиная от простого наблюдения до замеров сложнейшими приборами и целыми их комплексами. На фронте же основную работу по этому направлению вела разведка.

Войсковая разведка представляет собой тактическое звено огромного комплекса разведывательных мероприятий. Ее задача – вскрыть расположение войск и коммуникаций противника, получить данные о его боевых возможностях: «кто предупрежден – тот вооружен». Она является неотъемлемой частью любых военных действий.

Во время Великой Отечественной войны разведка велась постоянно и различными средствами. Это были скрытные проникновения в расположения врага, был и такой вид как разведка боем. О том, насколько плотно она велась, говорит, например, тот факт, что за один месяц в апреле 1943 года разведка корпуса провела более 245-ти вылазок. Большой удачей разведчиков была возможность захватить «языка», и чем выше был ранг пленённого противника, тем более ценными были его сведения. Однажды разведчикам в подбитом немецком танке попалась карта, на которой были подробно обозначены места нанесения ударов по нашим позициям.

Добывать такие сведения было по истине героической задачей – необходимо было не только наблюдение за противником, но и непосредственное углубление в его боевые порядки: «*… майор Карачун не только организовал тщательное наблюдение за противником, но и попытался переправить через Днепр еще несколько небольших разведгрупп*» [1]. Там же читаем: «*Отважные разведчики прорвались в ближний тыл врага и выявили расположение его огневых средств на этом направлении. Под сильным огнем противника … удалось к вечеру переправиться обратно*» [1]. Важные сведения доставляла и разведка средствами авиации.

Все эти сведения поступали в полк, получивший боевое задание, в оперативный отдел дивизии. Согласно данным строился план проведения операции, необходимая перестановка войск, снабжение огневыми средствами. Поэтому необходимым качеством информации была её достоверность. Как отмечает в своих мемуарах С. А. Андрющенко получаемые от разведчиков сведения сверялись и перепроверялись.

2. Передача информации различными средствами связи

Стоит поговорить о том, без чего невозможно было бы передать информацию, полученную разведкой, передать приказы по дивизии, отдать распоряжения – о связи.

Связь – важнейшая составляющая во время боевых действий. Без связи с руководством не будет никаких скоординированных действий и актуальных решений. В арсенале современных военных есть огромное множество способов связи. Спутниковая связь, закрытые мессенджеры, выделенные радиочастоты и даже отдельные телефонные линии. Но более семидесяти лет назад доброй половины современных способов связи попросту *не существовало*. Во время Великой Отечественной войны основными средствами связи являлись телефон, радио и вестовые. «*Начальник связи майор Д. Ф. Дроздов позаботился об устойчивой работе радиосредств, что было очень важно, поскольку войскам предстояло вести маневренные боевые действия, а проводная связь в таких условиях, тем более при форсировании крупной реки, недостаточно надежна*» [1]. Использовался и самолёт, когда в расположение войск прибывал представитель Ставки Верховного Главнокомандования для передачи строжайше секретного приказа о новом большом наступлении войск.

Отметим и такие средства как сигнальные ракеты: ими подавался сигнал к началу выступления. А вот пример совсем простого средства донесения информации, когда только его можно было использовать в отчаянной ситуации боя: «*Передать по цепи, – крикнул Хохряков, – что позади открытое пространство, и мы все погибнем, если начнем отход. Выход только один – вперед! Возьмем высоту броском! Артиллерия нам поможет*» [1].

Нельзя умолчать об отваге советских связистов, выполнявших немыслимые задачи в абсолютно нереальных условиях. В мемуарах описаны примеры героического самопожертвования солдат кабельно-шестовой роты, когда под огнём противника ремонтировали разбитые и повреждённые аппараты, по кускам собирали перебитые кабели. Именно благодаря тому, что связисты мужественно справлялись со своей работой, исполнением своего воинского долга могли заниматься и штабные офицеры.

3. Аналитико-синтетическая переработка информации в штабах

Штаб был и остается важнейшим звеном управления Вооруженными Силами, как в боевой обстановке, так и в мирное время. В его задачи входят: разработка оперативных и мобилизационных планов, контроль боевой подготовки армии, составление сводок и аналитических справок о положении войск, непосредственное руководство военными действиями.

Штаб – это, по сути, центр обработки информации и принятия решений. От качественного выполнения им своих функций зависят как общий уровень боевой подготовки войск, так и исход боевых действий. Как отмечено в [2], основными составляющими процесса информационно-аналитической деятельности помимо организационного и сбора данных являются

1) осмысление полученных данных фактов,
2) конструирование гипотез,
3) построение выводов,
4) изложение результатов.

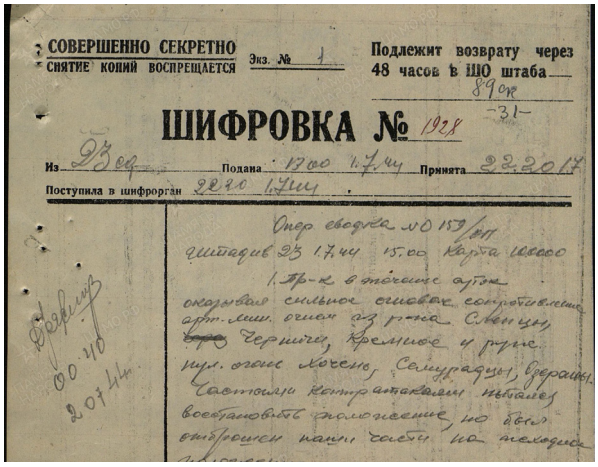
Чем более чётко и глубоко проведена оценка данных, рассмотрены возможные варианты, чем точнее предугаданы действия противника, тем более успешной будет предстоящая боевая операция.

Так однажды, во время подготовки наступления разведка донесла о двух бродах на территории, занятой немцами, которые можно было использовать. Противник пользовался только одним из них, так как второй был непроходим. В штабе решили использовать именно второй, поскольку противник не мог ждать их с этой стороны. Операция прошла успешно благодаря такому решению. При подготовке боя обсуждались вопросы взаимодействия частей, устанавливались варианты связи, оповещения, сигналы.

Работа в штабе представляла собой разбор полученной информации разными службами в пределах их компетенции и выработку вариантов действий путем общего согласования. «*Работа эта объемная и кропотливая. К ней привлекаются все специалисты: разведчики, артиллеристы, инженеры, связисты, политработники, тыловики. Каждый по своей линии готовит нужные сведения, представляет свои соображения и расчеты. Все это операторы затем анализируют, намечают наиболее целесообразные способы действия войск, мы их обсуждаем и докладываем комдиву. Отрабатывались различные схемы, составлялись боевые распоряжения, которые по мере готовности сразу же доводились до штабов частей. ... каждая пометка на карте – стрела, обведенный район, намеченные позиции – означала передвижение больших масс войск, сосредоточение техники, запасов, и любая неточность могла привести к очень нежелательным последствиям. Высокая штабная культура была и будет нужна, пока существует армия!*» [1].

4. Использование недостоверной информации и сокрытие информации

В период Великой Отечественной войны широко использовался такой вид деятельности как дезинформация противника. Это было мощным оружием, приносящим значительные успехи. Чтобы отвлечь противника от настоящего направления наступления наших частей, создавались ложные батареи, переправы, командно-наблюдательные пункты. «*Почти не маскируясь, огонь вели многие батареи. Расчет был простой: противник засечет наши огневые точки, нанесет их на карты и будет, как говорится, держать под прицелом. Мы же при отходе снимем орудия и минометы, а вместо них поставим макеты*» [1].


СОВЕРШЕННО СЕКРЕТНО
СНЯТИЕ КОПИЙ ВОСПРЕЩАЕТСЯ
Экз. № 1
Подлежит возврату через 48 часов в ШО штаба
ШИФРОВКА № 1928
Из ДЗод Подана 17.00 14.7.44 Принята 22.20 17
Поступила в шифрорган 22.40 17.7.44
Опер сводка №0159/оп
Рис. 1. Шифровка

Использовался и такой маневр, как отвлечение противника малыми силами с большим шумовым эффектом от расположения решительных сил в период наступления.

В то же время большое внимание в войсках уделялось сокрытию важной информации. В разговорах по телефону, радиосвязи требовалось соблюдать предельную осторожность. Виновные в пренебрежении этим правилом строго наказывались. Большую работу проводили шифровальные отделы при штабе.

5. Информация в политработе

Важное звено в войсках составляли политруки. Они первыми шли в атаку, увлекая за собой бойцов, проводили воспитательную, разъяснительную работу, поднимая боевой дух солдат,

утверждая веру в свои силы, силы своих товарищей и своей Родины. На коротких митингах до бойцов доводились слова командования, несшие в себе мощнейшие моральные установки: «*Днепр – не преграда, а мост к победе. Смелее преследуйте врага, вгрызайтесь в правый берег Днепра, не давайте врагу закрепиться!*» [1]. На таких митингах зачитывались тёплые и торжественные слова благодарности командования за героизм и отвагу, проявленную в сложной схватке с врагом.

Большую роль в поднятии боевого духа солдат играли армейские газеты, боевые листки, листки-молнии. В них давались сводки Совинформбюро, приказы Верховного Главнокомандующего, приказы о награждении, объявления благодарностей, описывались героические подвиги товарищей по оружию.

Рис. 2. Чтение боевого листка в окопах

Заключение

Многое не было здесь упомянуто, но и на основе приведённого материала можно видеть большое значение информационно-аналитической деятельности, проводившейся в войсках в годы Великой Отечественной войны. Мы видим, что работа с информацией так же важна, как и непосредственные боевые действия, что сбор и обработка информации не так просты, как может показаться, особенно в условиях боя.

Склоним же голову перед мужеством и героизмом всего советского народа: воинов, шедших в бой, не взирая ни на что, работников тыла, самоотверженно ковавших орудия Победы, простых людей, не сломавшихся под гнетом войны, всеми силами, кто чем мог, поддерживавших свою Родину, штабных и политработников, принимавших решения и поддерживавших нерушимый боевой дух бойцов.

Литература

1. *Андрющенко, С. А.* Начинали мы на Славутиче… : – М. : Воениздат, 1979. – 288 с.

2. *Захарова, И. С.* Основы информационно-аналитической деятельности : учеб. пособие / И. С. Захарова, Л. Я. Филиппова. – Киев : Центр учебной литературы, 2013. – 336 с.

Белов Никита Андреевич – студент 3-го курса кафедры математического моделирования Воронежского государственного университета. E-mail: alzacc@yandex.ru

Ильченко Антон Геннадиевич – студент 3-го курса кафедры математического моделирования Воронежского государственного университета. E-mail: antoniltch@yandex.ru

Адамова Римма Сергеевна (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры алгебры и математических методов гидродинамики Воронежского государственного университета. E-mail: adamova_rs@mail.ru

УДК 004.93’12

АЛГОРИТМ ЛИНГВИСТИЧЕСКОГО ИНДЕКСИРОВАНИЯ ИЗОБРАЖЕНИЙ

А. Э. Богомазова

Воронежский государственный университет

Введение

Количество графической информации, в особенности создаваемой пользователями интернета, непрерывно растет. Это связано с широким распространением мобильных устройств, оснащенных высококачественными цифровыми камерами, и ростом популярности сервисов публикации и распространения фотографий. Поиск в таком большом массиве информации – это сложная задача, требующая разработки эффективных алгоритмов индексирования.

Автоматическое лингвистическое индексирование изображений – это важная и актуальная проблема современной компьютерной графики. Многие исследования показывают, что разработка общего алгоритма, способного обрабатывать изображение и выдавать результат в виде лингвистического, то есть текстового описания – это сложная задача. Обычно такие алгоритмы строятся на методиках поиска изображений по содержанию (CBIR), которые позволяют искать изображения, основываясь, главным образом, на их внутренних характеристиках, вместо ключевых слов используется визуальная информация. Обычно в CBIR-системах шаблон изображения предоставляется пользователем, а в результате поиска возвращаются наиболее похожие изображения в соответствии с некоторой мерой сходства [1]. Однако такие системы не способны автоматически давать изображению текстовое описание, т. е. лингвистически индексировать. Многие исследователи пытались применить алгоритмы машинного обучения для лингвистического индексирования [2, 3]. Однако такие системы использовали алгоритмы обучения с учителем или трансдуктивные (полуавтоматические) методы поиска. Область обработки изображений стремительно развивается, объемы данных растут, а пользователи хотят получать результат быстро, поэтому все большую популярность приобретают алгоритмы автоматического лингвистического индексирования изображений.

Для моделирования отношений между изображениями во многих работах используются графы [4]. В простом графе изображения представляются вершинами, два похожих изображения соединяются ребром, а вес ребра вычисляется согласно некоторой мере сходства. Однако для представления отношений между изображениями простых графов недостаточно, поскольку необходимо брать в расчёт отношения не только между двумя вершинами, но и тремя или более вершинами, содержащими локальную классифицирующую информацию. Для учета этих фактов вводится понятие гиперграфа [5]. В данной статье представлен алгоритм автоматического лингвистического индексирования изображений, основанный на гиперграфе.

1. Понятие гиперграфа и его использование при оценке сходства изображений

Пусть $V = \{v_1, ..., v_n\}$ – конечное множество элементов. Гиперграфом [6, 7] на множестве V называется пара множеств $G = (V, E)$, где $E = \{e_1, ..., e_m\}$ – семейство подмножеств множества V такое, что $e_i \neq \varnothing$ для каждого $i = \overline{1, m}$ и $\bigcup_{i=1}^{m} e_i = V$. Элементы $v_1, ..., v_n$ множества V называются вершинами, а элементы $e_1, ..., e_m$ множества E – гиперребрами.

Гиперграф H является обобщением простого графа и представляет собой множество вершин, соединенных непрерывной кривой, заключающей внутри себя данные вершины при $|e_j| \geq 3$.

Гиперграф можно задать матрицей инцидентности $A = \{A(i,j)\}_{n \times m}$, в которой строки соответствуют вершинам $v_1, ..., v_n$, а столбцы – гиперребрам $e_1, ..., e_m$, при этом

$$A(i,j) = \begin{cases} 0, v_i \notin e_j, \\ 1, v_i \in e_j. \end{cases} \quad (1)$$

Взвешенным гиперграфом называется гиперграф $H = (V, E, W)$, в котором каждому гиперребру e_j поставлено в соответствие число $w(e_j) \in W$, называемое его весом.

В постановке задачи лингвистического индексирования изображений предполагается, что вершины – являются абстракцией изображений, а гиперребра – абстракцией категорий изображений, при этом вершине v_i ставится в соответствие гиперребро e_j с бинарной точностью $A(i,j) \in \{0,1\}$. Следовательно, относительное сходство между вершинами не учитывается, что приводит к потере информации, и, тем самым, снижает эффективность поисковой процедуры. Для того чтобы решить проблему этого ограничения вводится понятие нечеткого гиперграфа [5].

Предположим, что с использованием некоторой меры сходства вычисляется матрица сходства $S = \{S(i,j)\}_{n \times n}$ размерности $|V| \times |V|$, элементы которой $S(i,j) \in [0,1]$ оценивают сходство i-го и j-го изображений, сопоставленных i-й и j-й вершинам графа соответственно. Рассматривая каждую вершину графа v_i как центроиду, определим ее k-ближайших соседей и получим соответствующее гиперребро гиперграфа. Таким образом, мощность ребра в конструкции получается равной $(k+1)$. В этом случае элементы матрицы инцидентности A нечеткого гиперграфа определяются следующим образом:

$$A(v,e) = \begin{cases} S(u,v), v \in e, \\ 0, v \notin e. \end{cases} \quad (2)$$

Согласно этой формуле, вершина v «эластично» сопоставлена гиперребру e с учетом сходства $S(u,v)$ между u и v, где u – центроида гиперребра e.

2. Мера сходства изображений

Для вычисления меры сходства изображений могут быть использованы любые признаки изображений. Все алгоритмы поиска по содержанию можно разделить на классы в зависимости от характеристики, которую использует тот или иной алгоритм: поиск по цвету, по текстуре, по форме (рис. 1). Каждый из этих классов, в свою очередь, делится на подклассы по типу алгоритма построения вектора признака. Некоторые исследователи выделяют в отдельный класс пространственные признаки изображений. Под пространственными признаками понимают признаки, отражающие пространственное расположение на изображении областей, однородных по какой-либо характеристике: одного цвета, с однородной текстурой, распознанный объект. Можно сказать, что это признаки одного из классов (цвет, текстура, форма) с дополнительной информацией о пространственном расположении.

Очевидно, что наилучший результат может быть получен путем комбинирования этих признаков. Для выделения признаков изображений используются дескрипторы.

Дескриптором (вектором признаков) называется набор количественных параметров, описывающих характеристики изображения, например, такие как цвет, текстуру и т. д. Вектора признаков принимают значения в пространстве признаков. Если на таком пространстве задать меру, то можно сравнивать изображения друг с другом, вычисляя расстояние между соответствующими векторами признаков.

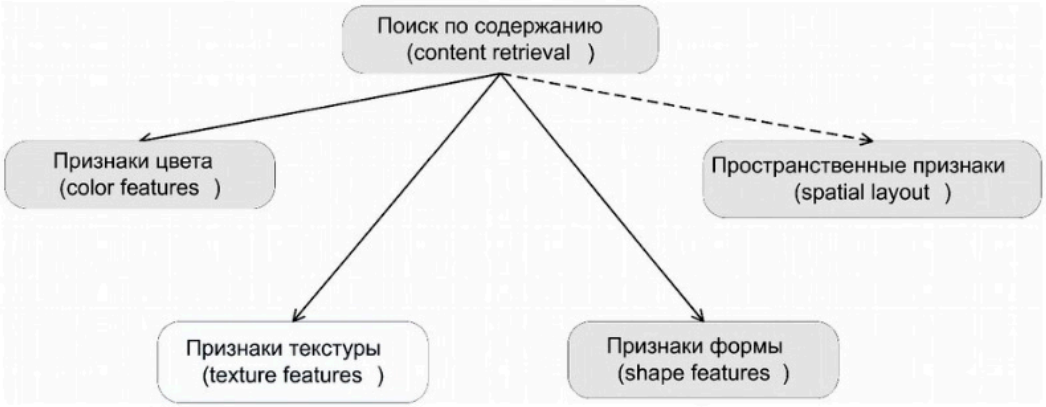


Рис. 1. Классификация методов поиска изображений

Можно использовать известные дескрипторы SIFT [8], OpponentSIFT, SURF, C-SIFT, RGB-SIFT [9] и HOG [10]. Например, SIFT-признаки локальны и основываются на проявлениях объекта в конкретных особых точах. Они инвариантны к изменению масштаба изображения и вращению. Также они стабильны к изменениям в освещении, шумам и небольшим изменениям точки наблюдения. Кроме этих свойств, они высоко различимы, относительно легко извлекаются и позволяют идентификацию объекта с малой вероятности ошибки.

Методы извлечения признаков, набор признаков отличаются в зависимости от дескриптора. Извлечение признаков с помощью разных дескрипторов помогает достигать наибольшей точности при оценке расстояний между изображениями.

Для расчёта расстояний между изображениями по каждому из извлеченных векторов признаков используется косинусная мера вида:

$$similarity = \frac{A * B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}, \quad (3)$$

где A и B – это векторы признаков размерности n.

После того как сходство по каждому дескриптору получено, вычисляется мера сходства S между двумя изображениями.

$$S(u,v) = \exp(-\frac{1}{k}\sum_{i=1}^{k}\frac{similarity_k}{\overline{similarity}}), \quad (4)$$

где k – количество извлеченных векторов признаков, $\overline{similarity}$ – среднее значение сходства.

3. Алгоритм лингвистического индексирования изображений

Общая схема поиска и индексирования изображений представлена на рис. 2.

Разделим наш алгоритм на два этапа: оффлайн фазу, в рамках которой осуществляется построение гиперграфа на основе существующих в базе данных изображений, и онлайн фазу лингвистического индексирования нового изображения.

Оффлайн фаза включает в себя следующие несколько этапов:

1) *Формирование исходной информации – коллекции изображений*. В нашем случае коллекция исходных изображений должна быть лингвистически проиндексирована, то есть каждому изображению должно быть дано текстовое описание.

2) *Определение меры сходства между исходными изображениями*. Для этого необходимо сначала выделить векторы признаков для каждого из изображений, а затем провести оценку

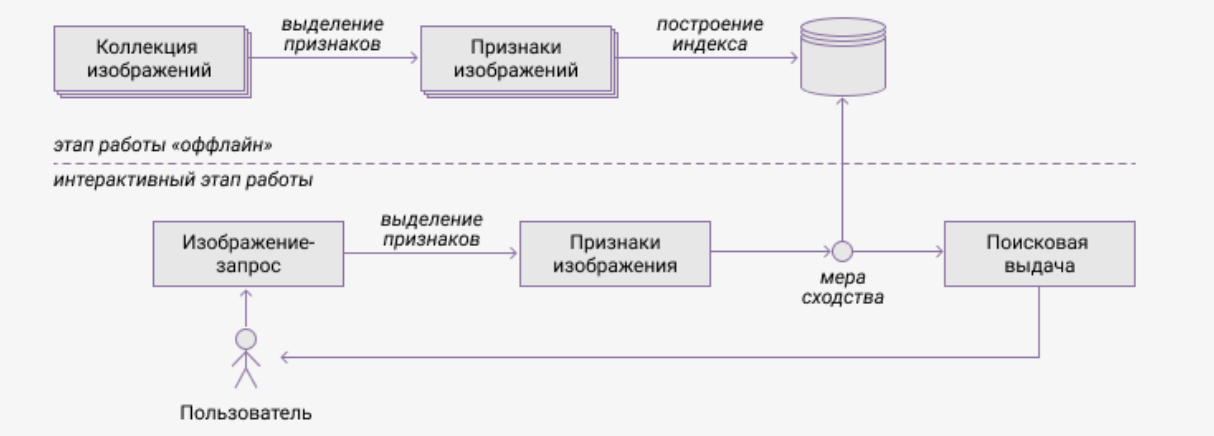


Рис. 2. Общая архитектура CBIR системы

сходства. В представленном алгоритме для выделения признаков используется набор различных дескрипторов, а для вычисления меры сходства используется косинусная мера.

3) *Упорядочивание и группировка изображений по степени их сходства*. Данный этап алгоритма реализуется на основе нечеткого гиперграфа. Таким образом, каждое гиперребро содержит набор вершин, то есть схожих изображений. Оценка меры сходства внутри каждого гиперребра достигается при помощи использования именно нечеткого гиперграфа. В результате каждому гиперребру можно сопоставить лингвистический индекс (набор текстовых описаний) входящих в него вершин.

На этом оффлайн фаза завершается, в результате получена упорядоченная и проиндексированная база изображений, представленная нечетким гиперграфом.

Теперь рассмотрим онлайн фазу.

1) На первом этапе пользователь предоставляет на вход изображение без какого-либо лингвистического описания. В результате работы алгоритма пользователь должен получить текстовое описание введенного изображения.

2) После того, как изображение получено, необходимо провести для него процедуру выделения признаков, по аналогии с тем, как это было сделано для оффлайн фазы. В результате будет получен набор векторов признаков входного изображения.

3) На этом этапе проводится оценка меры сходства между входным изображением и изображениями, представленными центроидами нечеткого гиперграфа. Таким образом, оценка производится для гораздо меньшего количества изображений, что значительно ускоряет выполнение. В результате находится гиперребро, центроида которого наиболее схожа с входным изображением.

4) на этом этапе извлекается лингвистический индекс для найденного на предыдущем этапе гиперребра. Если текстовое описание вполне однозначно, то есть гиперребру сопоставлено менее четырех текстовых описаний, то алгоритм завершает свою работу и выдает пользователю результат в виде текстового описания входного изображения. Если же гиперребру сопоставлено много различных текстовых описаний, проводится дополнительный поиск по вершинам гиперребра для нахождения изображений с максимальной мерой сходства к входному. На основании лингвистических индексов этих ближайших соседей выдается лингвистический индекс заданного на вход изображения.

Таким образом, алгоритм гораздо менее трудозатратен засчёт использования гипергрфа, так как сравнение мер сходства осуществляется лишь с изображениями центоид, и, при необходимости, с изображениями лишь одного гиперребра графа.

Заключение

В данной статье был представлен алгоритм автоматического лингвистического индексирования изображений, в котором нечеткий гиперграф используется для описания отношений релевантности между вершинами (изображениями). Каждое изображение рассматривалось как вершина-центроида и с помощью матрицы сходства, построенной на основе свойств изображений, помещалось вместе с k-ближайшими вершинами внутрь гиперребра. Таким образом, задача поиска среди всех изображений свелась к поиску среди центорид, что позволяет значительно ускорить процесс индексирования.

Литература

1. *He, X.* Learning and inferring a semantic space from user's relevance feedback for image retrieval / X. He, W.-Y. Ma, O. King, M. Li, H. Zhang – ACM MULTIMEDI, 2002. – 10 с.

2. *Minka, T. P.* Interactive Learning Using a Society of Models / T. P. Minka, R. W. Picard. – Pattern Recognition, 1997. – 565 с.

3. *Wang, J. Z.* Visual Similarity, Judgmental Certainty and Stereo Correspondence / J. Z. Wang, M. A. Fischler. – Proc. DARPA Image Understanding Workshop, 1998. – С. 1237–1248.

4. *Воробжанский, Н. Н.* Алгоритмы поиска изображений по содержанию с использованием нечеткого гиперграфа: системный анализ и информационные технологии / Н. Н. Воробжанский. – Вестник ВГУ, 2005. – С. 85–91.

5. *Берштейн, Л. С.* Нечеткие графы и гиперграфы / Л. С. Берштейн, А. В. Боженюк. – Москва : Научный мир, 2005. – 256 с.

6. *Емеличев, В. А.* Глава XI: Гиперграфы / В. А. Емеличев, О. И. Мельников, В. И. Сарванов, Р. И. Тышкевич // Лекции по теории графов. – М. : Наука, 1990. – С. 298–315.

7. *Зыков, А. А.* Гиперграфы / А. А. Зыков // Успехи математических наук . – 1974. – № 6 (180).

8. *Lowe, D.* Object recognition from local scale-invariant features / D. Lowe. – ICCV, 2009.

9. *Van de Sande, K. E. A.* Evaluating color descriptors for object and scene recognition / K. E. A. Van de Sande, T. Gevers, C. G. M. Snoek. – IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010.

10. *Dalal, N.* Histograms of oriented gradients for human detection / N. Dalal, B. Triggs. – CVPR, 2005.

Богомазова Алина Эдуардовна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: lina.a.b@yandex.ru

Леденева Татьяна Михайловна (научный руководитель) – д-р техн. наук, профессор, зав. кафедрой вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: ledeneva-tm@yandex.ru

УДК 004.75

АНАЛИЗ МЕТОДОВ ОПТИМИЗАЦИИ ОБРАБОТКИ БОЛЬШИХ ДАННЫХ НА SPARK STREAMING

Д. О. Васильев, Е. В. Трофименко

Воронежский государственный университет

Введение

Обработка большого массива данных в реальном времени предоставляет широкие возможности для использования в современных системах. При этом для быстрой обработки данных необходима их оптимизация, так как она помогает сделать различные процессы более эффективными при тех же затрачиваемых ресурсах. Одной из популярнейших комбинаций для этого является тандем Apache Kafka и Spark Streaming, где Kafka – создает поток пакетов входящих сообщений, а Spark Streaming обрабатывает эти пакеты через заданный интервал времени. Ею пользуются такие крупные компании, как: Yandex, Ebay, Amazon, Nasa и другие.

1. Описание проблемы

Spark streaming – компонент Spark, применяемый для обработки потоковых данных. Модуль Spark Streaming построен с применением «микропакетной» архитектуры (micro-batch architecture), когда поток данных интерпретируется как непрерывная последовательность маленьких пакетов данных. Spark Streaming принимает данные из разных источников и объединяет их в небольшие пакеты. Новые пакеты создаются через регулярные интервалы времени. В начале каждого интервала времени создается новый пакет, и любые данные, поступившие в течение этого интервала, включаются в пакет. В конце интервала увеличение пакета прекращается. Размер интервала определяется параметром, который называется интервал пакетирования (batch interval). При применении данной технологии обработки данных с разных потоков, данные могут приходить не постоянным потоком, а скачками, из-за чего могут возникнуть самые разнообразные проблемы, например, возникновение ошибки при переполнении памяти или недостаточная эффективность. Для примера будем рассматривать систему, которая способна обрабатывать 30 событий в секунду.

1.1. Проблема возникновения ошибки «OutOfMemory»

Свойства Spark настраиваются с помощью параметров приложения для каждой системы и задачи индивидуально. И если пренебречь подбором параметров, то можно получить ошибку переполнения памяти. Как видно на рис. 1, на первой секунде суммарное количество событий с каждого входящего потока превосходит максимальное, и из-за этого приложение останавливается с ошибкой «OutOfMemory».

1.2. Проблема в недостаточной эффективности

Для того чтобы не было вышеописанной проблемы, нужно указать значение для параметра «maxRatePerPartition», которое отвечает за количество событий с каждого входящего потока. Данное значение можно вычислить по формуле:

$$maxRatePerPartition = \left\lfloor \frac{maxRateAllPartitions}{numberOfAllPartitions} \right\rfloor = \left\lfloor \frac{30}{4} \right\rfloor = \lfloor 7.5 \rfloor = 7.$$

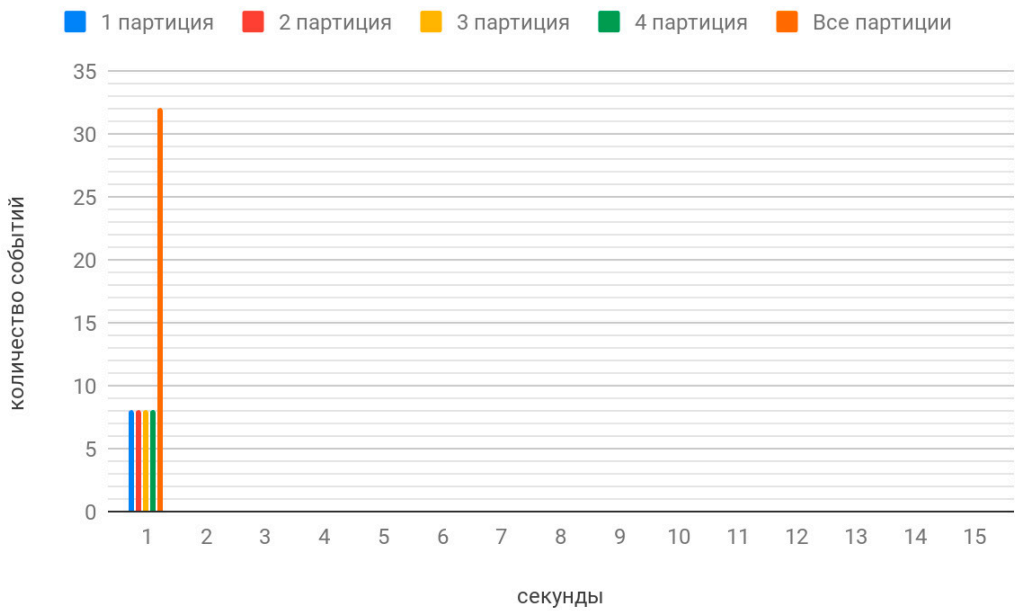


Рис. 1. Диаграмма обработки событий

На рис. 2 видно, что на 1 секунде со всех входящих потоков принято по 7 событий и система выполнилась без ошибки. Но начиная с 11 секунды активен один входящий поток и скорость обработки равна 7 событий в секунду, а система рассчитана на 30. В этом и заключается проблема в недостаточной эффективности.

К сожалению, с помощью стандартных средств, данную проблему исправить невозможно.

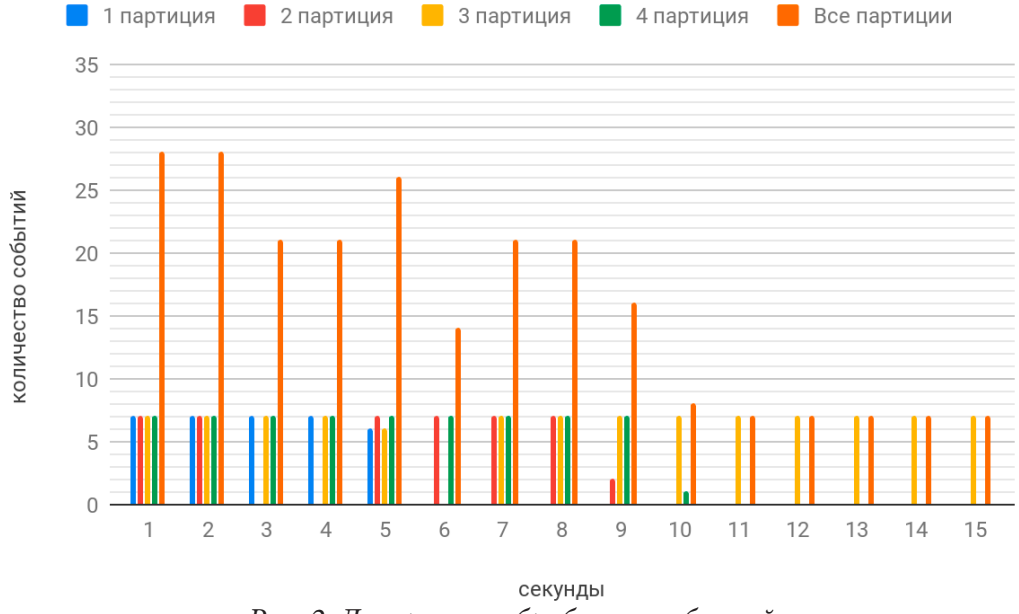


Рис. 2. Диаграмма обработки событий

2. Решение

Решением проблемы является система, которая способна автономно отслеживать ин-декс последнего события на каждом входящем потоке, а также до какого индекса события обработаны. На основе собранных данных, система должна решать нужно ли перезапускать потоковую обработку с новыми параметрами для оптимизации. На рис. 3 можно увидеть данную систему в виде схемы.

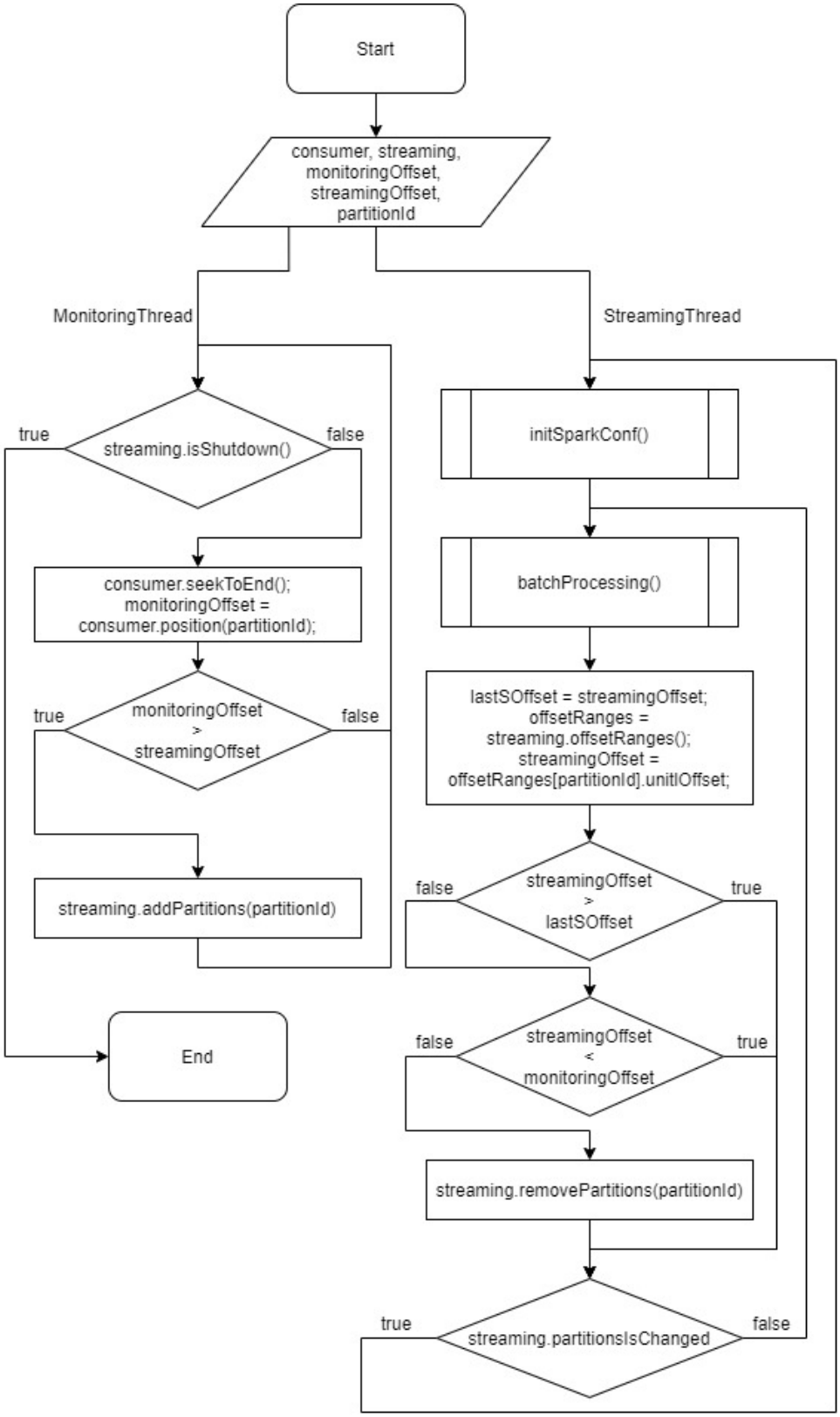


Рис. 3. Схема решения

На рис. 3 видно, что есть два потока «MonitoringThread» и «StreamingThread», каждый рассмотрим более подробно.

«MonitoringThread» – поток выполнения для отслеживания новых событий. В данном потоке происходит получение позиции последнего события и сохраняется в переменную

«monitoringOffset». Далее если позиция подготовленного больше позиции последнего обработанного события, то данный входящий поток должен быть в потоковой обработке. Данный алгоритм повторяется пока выполняется поток «StreamingThread».

«StreamingThread» – поток выполнения, в котором происходит обработка данных. Сначала запускается инициализация свойств Spark. Далее происходит обработка пакета данных. После обработки позиция последнего обработанного события присваивается перемен-ной «streamingOffset». Если не были обработаны события и «monitoringOffset» не больше «streamingOffset», то нужно исключить данный входящий поток из потоковой обработки. Если был изменен список входящих потоков, то повторяем алгоритм с момента инициализации, иначе с обработки следующего пакета данных.

Заключение

Результат работы системы представлен на рис. 4.

Анализируя результат работы можно видеть, что на каждой секунде система исполь-зует максимальное количество ресурсов системы. На рис. 3 было обработано 239 событий за 15 секунд, а на рис. 4 уже 446, что почти в 2 раза больше.

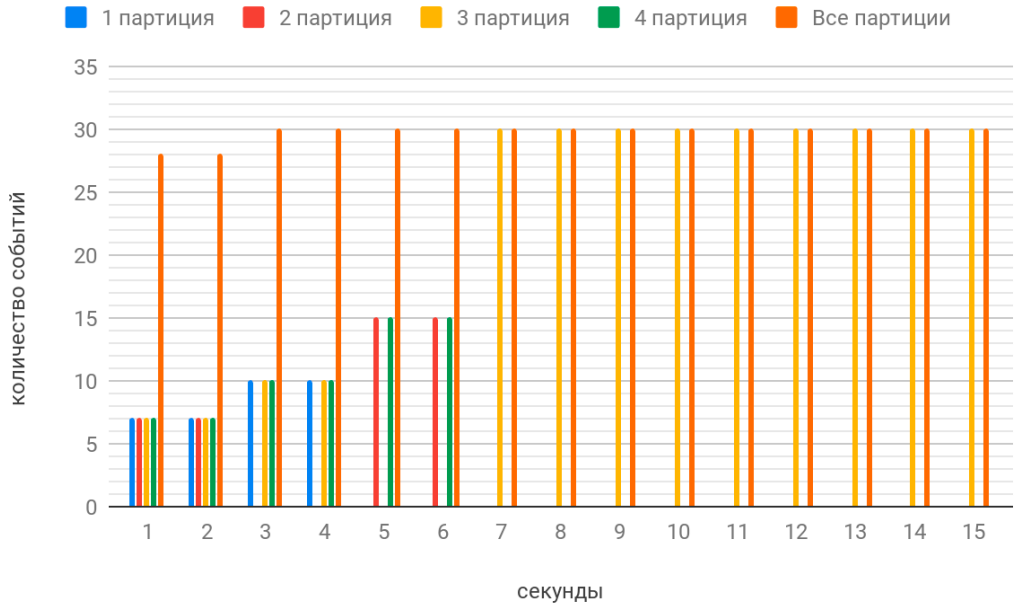


Рис. 4. Диаграмма обработки событий

Литература

1. *Карау, Х.* Эффективный Spark. Масштабирование и оптимизация / Х. Карау, Р. Уоррен. – Санкт-Петербург : Изд-во Питер, 2018 – 352 с.

2. Apache Spark™ – Unified Analytics Engine for Big Data. – Режим доступа: https://spark.apache.org/

Васильев Дмитрий Олегович – магистрант 2-го года обучения кафедры МО ЭВМ Воронежского государственного университета. E-mail: ceh4top@gmail.com

Трофименко Елена Владимировна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры МО ЭВМ Воронежского государственного университета. E-mail: evtrof@gmail.com

УДК 004.932.4

БЕСШОВНОЕ НАЛОЖЕНИЕ ИЗОБРАЖЕНИЙ С ПОМОЩЬЮ УРАВНЕНИЙ ПУАССОНА

И. А. Веселов

Воронежский государственный университет

Введение

В задачах машинного зрения и автоматизированной обработки изображений зачастую встречается задача бесшовного наложения изображений. В статье описывается решение данной задачи с использованием системы дифференциальных уравнений Пуассона с граничными условиями Дирихле, которая описывает Лапласиан неизвестной функции на области изображения.

1. Уравнения для решения задачи

На рис. 1 продемонстрированы обозначения:

$A \in R^2$ — замкнутое множество, является областью определения изображения;

B — замкнутое подмножество A с границей ∂B;

f — известная скалярная функция на $A \setminus \text{int}(B)$, описывающая изображение, на которое накладываем другое изображение;

g — неизвестная скалярная функция на $\text{int}(B)$;

h — известная функция на $\text{int}(B)$, описывающее накладываемое изображения;

v — направляющее векторное поле.

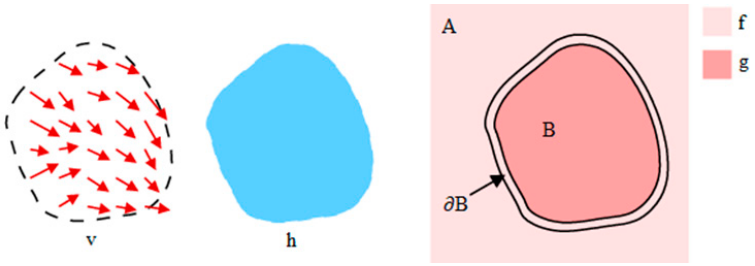


Рис. 1. Обозначения

Интерполяция g функции f на B определяется решением следующей задачи минимизации:

$$\min_g \iint_B |\nabla g|^2, g|_{\partial B} = f|_{\partial B}, \tag{1}$$

где $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ – оператор градиента. Решение данной задачи также удовлетворяет уравнению Лапласа с граничными условиями Дирихле:

$$\Delta g = 0, g|_{\partial B} = f|_{\partial B}, \tag{2}$$

где $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ – оператор Лапласа.

Данную задачу минимизации необходимо модифицировать с использованием направляющего поля v, чтобы сохранить накладываемое изображение:

$$\min_g \iint_B |\nabla g - v|^2, g|_{\partial B} = f|_{\partial B}. \tag{3}$$

Решение задачи (3) удовлетворяет следующему уравнению Пуассона с граничными условиями Дирихле:

$$\Delta g = \operatorname{div} v, g\big|_{\partial B} = f\big|_{\partial B}, \tag{4}$$

где $\operatorname{div} = \frac{\partial}{\partial x} + \frac{\partial}{\partial y}$ – оператор дивергенции.

В качестве направляющего поля v можно взять градиентное поле функции h, которое показывает направление наибольшего возрастания значения цвета.

Для задачи бесшовного наложения для каждого цветового канала составляется система уравнений в виде (4) и решается независимо от других систем.

2. Дискретное решение системы

Для дискретизации уравнения (4) моАжно использовать сетку пикселей изображения. Тогда задачу можно записать в следующем виде:

$$\min_{g} \sum_{\substack{(x,y)\in B,(x+dx,y+dy)\in B \\ dx\in\{-1,1\},dy\in\{-1,1\}}} (g(x,y) - f(x+dx,y+dy) - g(x+dx,y+dy) - v_{(x,y)(x+dx,y+dy)}), \tag{5}$$

где $v_{(x,y)(x+dx,y+dy)}$ – проекция $v(\frac{(x,y)+(x+dx,y+dy)}{2})$ на вектор $((x,y),(x+dx,y+dy))$.

Решение задачи (5) удовлетворяет следующим линейным уравнениям:

$$g(x,y) - \sum_{(dx,dy):(x+dx,y+dy)\in\partial B} f(x+dx,y+dy) - \sum_{(dx,dy):(x+dx,y+dy)\in\operatorname{int}(B)} g(x+dx,y+dy) - \\ -\left|\nabla v(x,y)\right|, \forall (x,y)\in \operatorname{int}(B), \tag{6}$$

где $\left|\nabla v(x,y)\right| = 4h(x,y) - h(x+1,y) - h(x-1,y) - h(x,y+1) - h(x,y-1).$

Заключение

Представленный метод возможно использовать для решения задачи бесшовного наложения изображений на компьютере. Составленные системы могут быть решены одним из известных методов решения разреженных систем линейных уравнений, например, методом Якоби.

Литература

1. *Гонсалес, Р. С.* Цифровая обработка изображений / Р. С. Гонсалес, Р. Е. Вудс. – 3-е изд., исправл. и доп. – Москва : Изд-во Техносфера, 2012. – 1104 с.

2. Загарян, Ю. А. Компьютерная графика в практических приложениях / Ю. А. Загарян, Е. В. Загарян. – Таганрог : Изд-во ТТИ ЮФУ, 2009. – 255 с.

3. *Цисарж, В. В.* Математические методы компьютерной графики / В. В. Цисарж, Р. И. Марусик. – Киев : Изд-во Факт, 2004. – 464 с.

4. *Павлидс, Т.* Алгоритмы машинной графики и обработки изображений / Т. Павлидс. – Изд-во Радио и связь, 1986. – 400 с.

5. *Прэтт, У.* Цифровая обработка изображений. Книга 1 / У. Прэтт. – Москва : Изд-во Мир, 1982. – 311 с.

6. *Прэтт, У.* Цифровая обработка изображений. Книга 2 / У. Прэтт. – Москва : Изд-во Мир, 1982. – 479 с.

7. Применение преобразования Пуассона для бесшовного наложения изображений: http://habr.com/ru/post/213403/. – (Дата обращения: 19.04.2020).

8. *Перез, П.* Poisson Image Editing / П. Перез, М. Гангнет, А. // ACM Transactions on Graphics. – 2003.– С. 313-318.

Веселов Илья Александрович – магистрант 1-го года обучения кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: ves.ilya@mail.ru

Горбенко Олег Данилович (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: oleg_dan@mail.ru

УДК 539.3

ОПРЕДЕЛЕНИЕ НАПРЯЖЕННОГО СОСТОЯНИЯ УПРОЧНЯЮЩЕГОСЯ ДИСКА ПОД ДЕЙСТВИЕМ ТЕМПЕРАТУРЫ

М. М. Вислогузова

Воронежский государственный университет

Введение

Определению напряжений и деформаций в дисках и пластинах посвящено множество работ, в том числе и [1, 3–7].

В данной работе, решается задача о точечном нагреве диска. Тепловой источник действует в центре диска из анизотропного упрочняющегося упругопластического материала. Такой процесс может возникать при точечной контактной сварке. Решение проводилось с помощью метода малого параметра. В результате, были получены: напряжения в упругой и пластической областях, радиус границы раздела зон упругого и пластического деформирования.

1. Постановка задачи

Пусть тонкий диск находится в рамках плоского напряженного состояния, все механические и тепловые константы материала не зависят от температуры.

Вследствие осевой симметрии все искомые величины будут зависеть только от расстояния r до точечного источника.

Поскольку в окрестности теплового источника температура теоретически неограниченно высока, там сразу возникает пластическая область. Пусть $r = R$ – граница этой области. Внешнюю границу диска обозначим $r = b$.

Запишем систему уравнений осесимметричного плоско напряженного состояния упругопластического диска в цилиндрической системе координат (r, θ, z):

– уравнение равновесия

$$\frac{\partial \sigma_r}{\partial r} = \frac{\sigma_\theta - \sigma_r}{r}, \tag{1.1}$$

где σ_r, σ_θ – компоненты тензора напряжений в цилиндрической системе координат;

– соотношения Коши

$$e_r = \frac{du}{dr}, \; e_\theta = \frac{u}{r}, \tag{1.2}$$

где e_r, e_θ – компоненты тензора деформаций в цилиндрической системе координат, u – компонента вектора перемещений в цилиндрической системе координат;

– связь полных деформаций с пластической и упругой составляющими

$$e_r = e_r^p + e_r^e, \; e_\theta = e_\theta^p + e_\theta^e, \; e_z = e_z^p + e_z^e. \tag{1.3}$$

Функцию нагружения выберем в форме, предложенной Ишлинским и Прагером, аналогично [5]:

$$(\sigma_r - \sigma_\theta - c(e_r^p - e_\theta^p))^2 + (\sigma_r - c(e_r^p - e_z^p))^2 + (\sigma_\theta - c(e_\theta^p - e_z^p))^2 = 2k^2, \tag{1.4}$$

где e_r^p, e_θ^p, e_z^p – компоненты тензора пластических деформаций, c – коэффициент упрочнения, k – предел текучести.

Ассоциированный закон пластического течения тогда будет иметь вид:

$$\begin{aligned} de_r^p &= 2d\lambda(\sigma_r - \sigma_\theta - c(2e_r^p - e_\theta^p - e_z^p)), \\ de_\theta^p &= 2d\lambda(\sigma_\theta - \sigma_r - c(2e_\theta^p - e_r^p - e_z^p)), \\ de_z^p &= -2d\lambda(\sigma_r + \sigma_\theta + c(2e_z^p - e_\theta^p - e_z^p)), \end{aligned} \tag{1.5}$$

где $d\lambda$ – малый скалярный положительный множитель.

Упругие деформации связаны с напряжениями законом Гука в форме:

$$\begin{aligned} \sigma_r &= \frac{2G}{1-\nu}[e_r^e + \nu e_\theta^e - (1+\nu)\alpha T], \\ \sigma_\theta &= \frac{2G}{1-\nu}[e_\theta^e + \nu e_r^e - (1+\nu)\alpha T], \end{aligned} \tag{1.6}$$

где G – модуль сдвига, ν – коэффициент Пуассона, α – коэффициент линейного температурного расширения, $T(r)$ – температура, которая определяется из решения уравнения теплопроводности см. [1], [4].

Граничные условия:

$$\sigma_r\big|_{r=b} = 0. \tag{1.7}$$

Условия сопряжения:

$$[\sigma_r]|_{r=R} = [\sigma_\theta]|_{r=R} = [u]|_{r=R} = 0. \tag{1.8}$$

Далее все величины, имеющие размерность напряжений, отнесем к k. Для безразмерных величин сохраним исходные обозначения.

2. Линеаризация

Для решения данной задачи, будем пользоваться методом малого параметра. Согласно этому методу:

$$\begin{aligned} &\sigma_r = \overset{0}{\sigma}_r + \delta\sigma_r', \ \sigma_\theta = \overset{0}{\sigma}_\theta + \delta\sigma_\theta', \\ &e_r = \overset{0}{e}_r + \delta e_r', \ e_\theta = \overset{0}{e}_\theta + \delta e_\theta', \ e_z = \overset{0}{e}_z + \delta e_z', \\ &c = \overset{0}{c} + \delta c', \ \alpha = \overset{0}{\alpha} + \delta\alpha', \ \lambda = \overset{0}{\lambda} + \delta\lambda', \end{aligned} \tag{2.1}$$

где величины с индексом «0» относятся к нулевому приближению, а с индексом «'» – к первому.

Причем, будем считать, что:

$$\overset{0}{c} = \alpha' = 0 \Rightarrow c = \delta c', \ \alpha = \overset{0}{\alpha}. \tag{2.2}$$

Таким образом, для упругой области, в результате линеаризации, в нулевом приближении уравнения (1.1), (1.2), (1.6) сохраняют свой вид. Для первого приближения уравнения (1.1), (1.2) остаются прежними.

Для пластической области в нулевом приближении:

– условие пластичности:

$$\overset{0}{\sigma}_r^2 + \overset{0}{\sigma}_\theta^2 - \overset{0}{\sigma}_r \overset{0}{\sigma}_\theta = 1 \tag{2.3}$$

– ассоциированный закон:

$$\begin{aligned} d\overset{0}{e}_r^p &= 2d\overset{0}{\lambda}(\overset{0}{\sigma}_r - \overset{0}{\sigma}_\theta), \\ d\overset{0}{e}_\theta^p &= 2d\overset{0}{\lambda}(\overset{0}{\sigma}_\theta - \overset{0}{\sigma}_r), \\ d\overset{0}{e}_z^p &= -2d\overset{0}{\lambda}(\overset{0}{\sigma}_r + \overset{0}{\sigma}_\theta). \end{aligned} \tag{2.4}$$

Для пластической области в первом приближении:

– условие пластичности:

$$(\overset{0}{\sigma}_r-\overset{0}{\sigma}_\theta)(\sigma'_r-\sigma'_\theta-c'(\overset{0}{e}{}^p_r-\overset{0}{e}{}^p_\theta))+\overset{0}{\sigma}_r(\sigma'_r-c'(\overset{0}{e}{}^p_r-\overset{0}{e}{}^p_z))+\overset{0}{\sigma}_\theta(\sigma'_\theta-c'(\overset{0}{e}{}^p_\theta-\overset{0}{e}{}^p_z))=0. \qquad (2.5)$$

3. Решение в нулевом приближении

3.1. Решение в упругой области

Подставляя в уравнение равновесия (1.1) σ_r и σ_θ, выраженные из закона Гука с учетом температуры (1.6), и учитывая соотношения Коши (1.2):

$$\frac{\partial^2 \overset{0}{u}}{\partial r^2}+\frac{1}{r}\frac{\partial \overset{0}{u}}{\partial r}-\frac{\overset{0}{u}}{r^2}=(1+\nu)\overset{0}{\alpha}\frac{\partial T}{\partial r}. \qquad (3.1.1)$$

Решение уравнения (3.1.1) имеет вид:

$$\overset{0}{u}=\frac{A(t)}{r}+\frac{\overset{0}{\alpha}(1+\nu)}{r}\int Trdr. \qquad (3.1.2)$$

Из (3.1.2) по соотношениям Коши (1.2) были найдены полные деформации:

$$\begin{aligned}\overset{0}{e}_r&=-\frac{1}{r^2}(A(t)+\overset{0}{\alpha}(1+\nu)\int Trdr)+\overset{0}{\alpha}T(1+\nu),\\ \overset{0}{e}_\theta&=\frac{1}{r^2}(A(t)+\overset{0}{\alpha}(1+\nu)\int Trdr)\end{aligned} \qquad (3.1.3)$$

Полученные деформации подставили в закон Гука. Получили выражения для напряжений в упругой области в нулевом приближении:

$$\begin{aligned}\overset{0}{\sigma}_r&=-2G(\frac{1}{r^2}(A(t)+\overset{0}{\alpha}(1+\nu)\int Trdr)),\\ \overset{0}{\sigma}_\theta&=-\overset{0}{\sigma}_r-2GT\overset{0}{\alpha}(1+\nu).\end{aligned} \qquad (3.1.4)$$

3.2. Решение в пластической области

В пластической области, согласно [1]:

$$\overset{0}{\sigma}_r=\overset{0}{\sigma}_\theta=-1. \qquad (3.2.1)$$

Из ассоциированного закона (2.4) получим:

$$\frac{d\overset{0}{e}{}^p_r}{2\overset{0}{\sigma}_r-\overset{0}{\sigma}_\theta}=\frac{d\overset{0}{e}{}^p_\theta}{2\overset{0}{\sigma}_\theta-\overset{0}{\sigma}_r}. \qquad (3.2.2)$$

Учитывая (3.2.1), (3.2.2) можно записать в виде:

$$d\overset{0}{e}{}^p_r=d\overset{0}{e}{}^p_\theta \Rightarrow \frac{\partial}{\partial t}(\frac{\partial \overset{0}{\tilde{u}}}{\partial r}-\frac{\overset{0}{\tilde{u}}}{r})=0, \qquad (3.2.3)$$

где $\overset{0}{\tilde{u}}$ – компонента вектора перемещений в нулевом приближении в пластической области.

Общее решение (3.2.3):

$$\overset{0}{\tilde{u}}(r,t)=r(C(t)+D(r)). \qquad (3.2.4)$$

Используя условия сопряжения (1.8), получим:

$$A(t)=\frac{R^2}{2G}(-\overset{0}{\alpha}(1+\nu)\int Trdr). \tag{3.2.5}$$

Подставляя (3.2.5) в выражение для $\overset{0}{\sigma}_\theta$ из (3.1.4) и приравнивая –1, имеем:

$$G\overset{0}{\alpha}(1+\nu)T(R(t),t)=1. \tag{3.2.6}$$

Отсюда

$$T(R(t),t)=\frac{2}{G\overset{0}{\alpha}(1+\nu)}. \tag{3.2.7}$$

Из уравнений (3.2.4) и (3.2.7) получаем:

$$\overset{0}{\tilde{u}}(r,t)=\frac{r}{2G}. \tag{3.2.8}$$

Из (3.2.8) следует:

$$\overset{0}{e}{}^p_\theta=\overset{0}{e}{}^p_r=\frac{(1+2\nu)}{G\overset{0}{\alpha}(1+\nu)}-\overset{0}{\alpha}T. \tag{3.2.9}$$

4. Решение в первом приближении

4.1. Решение в пластической области

Из (2.3) и (2.5) получили связь между σ'_θ и σ'_r:

$$\sigma'_\theta=-\sigma'_r\frac{(2\overset{0}{\sigma}_r-\overset{0}{\sigma}_\theta)}{(2\overset{0}{\sigma}_\theta-\overset{0}{\sigma}_r)}+3c'\frac{(\overset{0}{e}{}^p_r\overset{0}{\sigma}_r+\overset{0}{e}{}^p_\theta\overset{0}{\sigma}_\theta)}{(2\overset{0}{\sigma}_\theta-\overset{0}{\sigma}_r)}. \tag{4.1.1}$$

Подставляя в (4.1.1) (3.2.1) и (3.2.9), получим:

$$\sigma'_\theta=-\sigma'_r+6c'(\frac{2(1+2\nu)}{G(1+\nu)}-\overset{0}{\alpha}T). \tag{4.1.2}$$

Выражение (4.1.2) подставили в уравнение равновесия. В результате данной подстановки получили дифференциальное уравнение для определения σ'_r:

$$r\frac{d\sigma'_r}{dr}+2\sigma'_r=\frac{6c'}{G(1+\nu)}((1+2\nu)-G\overset{0}{\alpha}(1+\nu)T). \tag{4.1.3}$$

Решение данного уравнения имеет вид:

$$\sigma'_r=3c'\left[\frac{(1+2\nu)}{G(1+\nu)}-\frac{2}{r^2}\int r\overset{0}{\alpha}Tdr\right]+C, \tag{4.1.4}$$

где C – неопределенная константа.

Подставив (4.1.4) в (4.1.1):

$$\sigma'_\theta=3c'\left[\frac{(1+2\nu)}{G(1+\nu)}+2\left(\frac{1}{r^2}\int r\overset{0}{\alpha}Tdr-\overset{0}{\alpha}T\right)\right]+C. \tag{4.1.5}$$

Для определения константы C воспользуемся условием:

$$\sigma'_r(0)=3c'(\frac{(1+2\nu)}{G(1+\nu)}-\overset{0}{\alpha}T). \tag{4.1.6}$$

(4.1.6) получено аналогично получению условия в [4].

4.2. Решение в упругой области

Решение в первом приближении для упругой области определяется согласно [2], [3]. В результате были получены соотношения для определения напряжений:

$$\sigma_r' = \frac{2G}{1-\nu}[C_1(1+\nu) - C_2\frac{1}{r^2}(1-\nu)],$$
$$\sigma_\theta' = \frac{2G}{1-\nu}[C_1(1+\nu) + C_2\frac{1}{r^2}(1-\nu)]. \quad (4.2.1)$$

Для определения констант C_1 и C_2 воспользуемся линеаризованным условием сопряжения (1.8).

Таким образом, в результате получены: напряжения в упругой и пластической областях, радиус границы раздела зон упругого и пластического деформирования для нулевого и первого приближения.

Литература

1. *Паркус, Г.* Неустановившиеся температурные напряжения / Г. Паркус. – Москва : Государственное издательство физико-математической литературы, 1963. – 252 с.

2. *Демидов, С. П.* Теория упругости / С. П. Демидов. – Москва : Высшая школа, 1979. – 432 с.

3. Тимошенко, С. П. Теория упругости / С. П. Тимошенко, Дж. Гудьер. – Москва : Наука, 1979. – 560 с.

4. *Артемов М. А.* Математическое моделирование механического поведения вращающегося диска / М. А. Артемов, А. П. Якубенко // Воронеж. гос. ун-т. – 2014. – № 1. – С. 30–38.

5. О механическом поведении упрочняющегося упругопластического диска под действием источника тепла / А. А. Афанасьев, К. К. Горностаев, А. В. Ковалев, А. С. Чеботарев // Вестник Томского гос. ун-та. Сер. Математика и механика. – 2017. – № 50. – С. 57–66.

6. Математическое моделирование состояния тонкого диска при тепловом и силовом воздействиях [Сетевое издание] / Артемов М. А., Барановский Е. С., Акиньшин В. В., Скорняков Н. С., Фатхудинов Д. Б. // Инженерный вестник Дона.-науч. журн. – 2019. – № 4. – Режим доступа: http://www.ivdon.ru/uploads/article/pdf/IVD_28_Artemov.pdf_4c0401978b.pdf. – (Дата поступления статьи: 07.06.2019).

7. О поведении упругопластического диска под действием теплового источника [Сетевое издание] / Бердзенишвили Г. Г., Артемов М. А., Барановский Е. С., Семка Э. В., Фатхудинов Д. Б. // Инженерный вестник Дона.-науч. журн. – 2018. – № 2. – Режим доступа: http://www.ivdon.ru/uploads/article/pdf/IVD_190_Berdzeneshvili_N.pdf_45978b1390.pdf– (Дата поступления статьи: 01.06.2018).

Вислогузова Мария Михайловна – студентка 4-го курса кафедры механики и компьютерного моделирования Воронежского государственного университета.
E-mail: visloguzova99@mail.ru

Ковалев Алексей Викторович (научный руководитель) – д-р физ.-мат. наук, проф., профессор кафедры механики и компьютерного моделирования Воронежского государственного университета. E-mail: kav-mail@mail.ru

УДК 004.9

ИССЛЕДОВАНИЕ ВЛИЯНИЯ НАСТРАИВАЕМЫХ ПАРАМЕТРОВ LSM-ДЕРЕВА НА ЕГО ПРОИЗВОДИТЕЛЬНОСТЬ

И. Б. Глазырин

Воронежский государственный университет

Введение

Стандартная реализация LSM-дерева (от Log-structured merge-tree – журнально-структурированное дерево со слиянием) описывается как структура данных, предназначенная для обеспечения недорогой индексации для файла с высокой частотой вставки и удаления записей на протяжении длительного периода времени [1]. Такая структура обеспечивает лучшие шаблоны доступа ввода-вывода для интенсивных рабочих нагрузок на запись данных, но в то же время – снижение производительности запросов на чтение.

Чтобы попытаться улучшить производительность, было разработано LSM-дерево, которое использует различные настраиваемые параметры. Эти параметры регулируют поведение LSM-дерева и позволяют ему хранить и читать данные с использованием различных техник. LSM-деревья распространены во многих современных системах и поэтому эта работа может найти применение в многочисленных сферах, например, классические системы баз данных или хранилище данных типа «ключ-значение».

1. Общая концепция

Базовую структуру LSM-дерева можно представить в виде двух уровней (рис. 1). Первый уровень, называемый C0, полностью находится в памяти, в то время как второй уровень, C1, находится на диске [2]. Когда первый уровень заполняется, данные записываются в новый файл на диске (рис. 2). Этот процесс повторяется по мере того, как поступает все больше и больше записей.

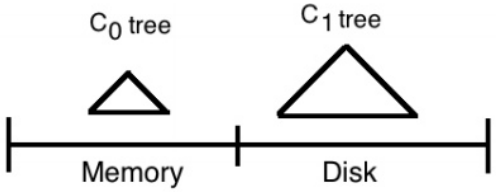


Рис. 1. Базовая структура LSM-дерева

Такая многоуровневая структура LSM-дерева снижает эффективность запросов на чтение. Если запрашивается чтение, потребуется проверить несколько уровней, и для каждого уровня потребуется одно чтение. Хотя изначально LSM-дерево представлено как структура данных, эффективная для записи, некоторые проектные решения могут быть изменены, чтобы уменьшить влияние этого дизайна на запросы чтения.

Возможность изменять настройки некоторых параметров будет положительно влиять на производительность. Например, увеличение размера уровня дерева, находящегося в памяти, уменьшение количества уровней и увеличение размера каждого уровня приведет к уменьшению среднего времени чтения. Это связано с тем, что большее количество данных, хранящихся в памяти, и меньшее количество уровней для хранения данных приведут к меньшему количе-

ству операций чтения с диска. Кроме того, по мере увеличения количества уровней производительность операций вставки будет уменьшаться, поскольку несколько уровней включают в себя множество более мелких структур данных, которые необходимо поддерживать. Это приведет к более частой передаче данных с одного уровня на другой.

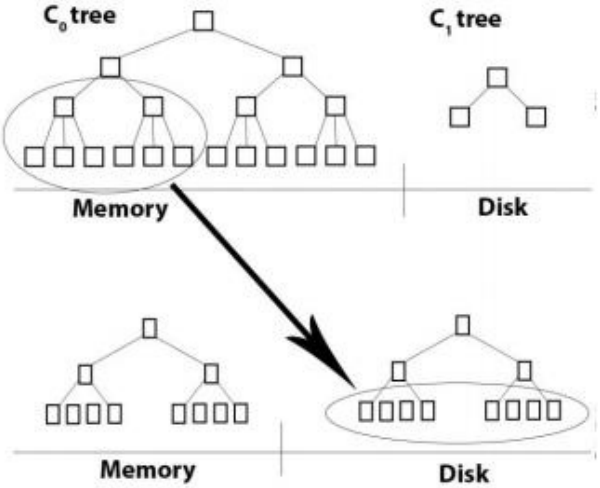


Рис. 2. Слияние данных в LSM-деревьях

2. Настраиваемые параметры LSM-дерева

В данной работе было реализовано LSM-дерево, которое предоставляет функции вставки, чтения, удаления и обновления. Оно было написано на языке C++ и использует парадигму объектно-ориентированного программирования (OOP) для включения трех основных классов. Класс LSMTree – это основной класс, который использует структуры данных, которые находятся как в памяти (класс LsmLevelMemory), так и на диске (класс LsmLevelDisk). Данное дерево использует различные настраиваемые параметры для оптимизации производительности.

Первый настраиваемый параметр называется is_read_Optimized. Это логическое значение, которое определяет, следует ли использовать для запросов на чтение реализованную версию фильтра Блума [3]. При большом количестве операций на чтение это уменьшит количество взаимодействий ввода / вывода, но потребует больше памяти для поддержки фильтра Блума.

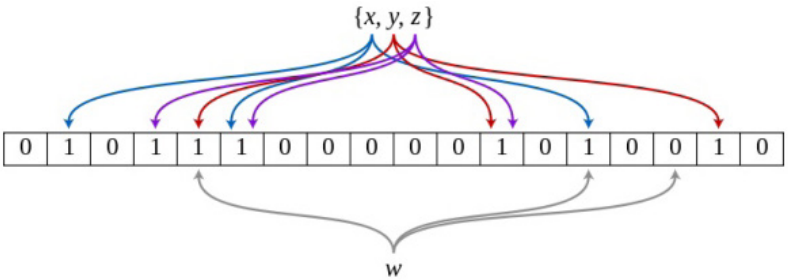


Рис. 3. Фильтр Блума

Второй настраиваемый параметр – m_node_size. Это целое число, которое представляет количество значений, которые будут сохранены в каждом узле. Общий подход заключается в том, что размер узла должен быть как можно меньше, чтобы не приносить слишком много данных с диска.

Структура данных, которая будет использоваться для первого уровня C0 в памяти, определяется третьим булевским параметром, c0DataStructure, который имеет опцию для B-дерева и другой для массива.

Настройка отношения между уровнями позволяет управлять балансом между производительностью чтения и записи проекта (табл. 1). Так, стоимость поиска зависит от количества уровней. Мы уменьшаем количество уровней, увеличивая соотношение размеров между уровнями. Мы могли бы добиться этого в серии экспериментов, установив параметр firstLevelFileSize (размер первого уровня) на 200 000 байт и уменьшив sizeBetweenLevels (разница между размерами уровней) с 4 до 2.

Таблица 1

Временные сложности для вставок и чтений несортированного массива и B-дерева

	Вставка	Чтение
Несортированный массив	O(1)	O(n)
B-дерево	O(log(n))	O(log(n))

Следующий параметр, num_levels, является целочисленным значением, которое устанавливает верхнюю границу общего количества уровней для LSM-дерева. Это позволяет выполнить отладку на этапе разработки, чтобы оценить правильность операций сжатия данных.

Однопоточное LSM-дерево может быть превращено в многопоточное, если для логического параметра threadedRollingMerge установлено значение true. Это заставляет процессы объединенного слияния выполняться в отдельном потоке, позволяя одновременно выполнять большее количество вставок. Это также включает рабочую очередь и реализацию пула потоков, которые позволяют одновременное чтение и удаление, чтобы позволить программе масштабироваться по мере роста числа потоков / ядер. Этот параметр полезен во время экспериментов, чтобы понять влияние распараллеливания на систему во время рабочих нагрузок, связанных с одновременным чтением, записью и объединением слияний.

3. Результаты исследования

Основным компьютером, использовавшимся для тестирования, был компьютер с процессором Intel® Core i9-9900K, 16 ГБ ОЗУ и жестким диском на 500 МБ. Размер используемых наборов данных варьировался от 20 000 случайных чисел до 256 миллионов случайных чисел. Рабочая нагрузка варьировалась с ориентированной на запись или чтение до сочетания двух при различных соотношениях. Чтобы протестировать различные сценарии, были добавлены рабочие нагрузки, представляющие собой повторяющиеся чтения значений (считывание одного и того же значения много раз) и обновления, за которыми следуют удаления и чтения.

3.1. Тип структуры данных для уровня, находящегося в памяти

В данной реализации LSM-дерева имеется параметр для изменения структуры данных уровня памяти C0 с B-дерева на вектор стандартной библиотеки C ++. Во время тестирования были проверены различные рабочие нагрузки, чтобы определить, есть ли польза от любой структуры данных. Чтение и запись протестированы в размерах от 50 тысяч до 500 тысяч значений, и в результате не было выявлено существенного различия. Принято решение использовать std::vector для окончательного тестирования. Это означало, что вставки можно было сделать более простым способом, используя метод push_back, в отличие от реализации B-дерева, которая должна была перенастраивать форму дерева после каждой вставки. Различные тесты также показали, что для небольших наборов потока данных отсортированный вектор может работать лучше, чем B-дерево.

3.2. Определение оптимального размера узла

Каждый уровень LSM-дерева содержит одну или несколько древовидных структур данных. Эти древовидные структуры данных содержат много узлов. Каждый узел содержит размер, который определяет, сколько значений может находиться внутри него. Насколько большим должен быть каждый узел дерева? Прочно установившаяся теория обычно предполагает, что базовые дисковые накопители используют блок фиксированного размера, и устанавливает размер блока дерева в соответствии с размером блока диска [5]. Другая теория предполагает, что оптимальный размер узла определяется в первую очередь задержкой доступа и пропускной способностью передачи, а также размером записи [6].

Данная реализация LSM-дерева содержит параметр для установки размера узла как для уровня, находящегося в памяти, так и для дисковых структур данных. После тестирования было определено, что оптимальный размер узла равен 20 как для вставки с 200000 значений (красная линия на рис. 4), так и для вставки с 800000 значениями (синяя линия на рис. 4).

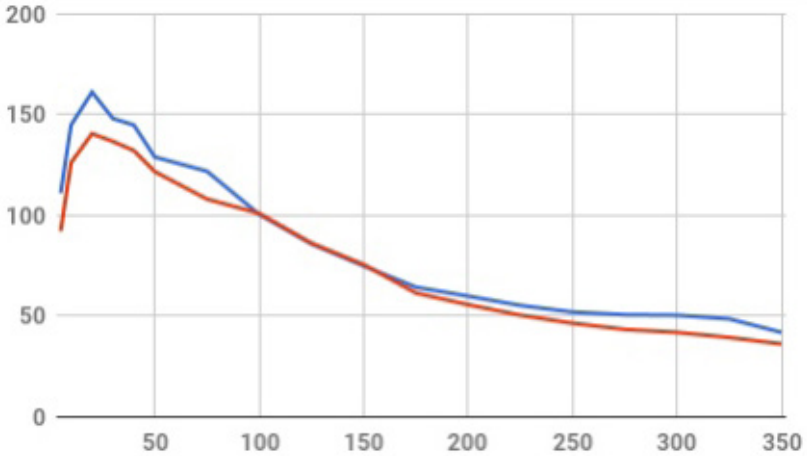


Рис. 4. Определение оптимального размера узла

3.3. Многопоточность

Для повышения производительности LSM-дерева в многоядерной системе используется многопоточность. В данном случае в отдельный поток можно перенести процесс слияния. Один поток выполняет все вставки в память до тех пор, пока не будет запрошено скользящее слияние (поскольку уровень в памяти достиг определенного предела). Скользящее слияние затем выполняется в отдельном потоке, что, в свою очередь, будет выполняться одновременно с заполнением первого уровня до тех пор, пока не будет запрошено очередное слияние. Этот процесс позволяет уровню в памяти быстро поглощать большее количество вставок, не ожидая завершения процесса слияния в однопоточном случае (рис. 5).

Заключение

В результате проделанной работы можно сделать вывод, что настраиваемые параметры по-разному могут влиять на производительность LSM-дерева. Так, повышение числа потоков с одного до двух повышает скорость вставок в секунду примерно в два раза, тип структуры данных (std:vector или B-дерево) не существенно влияет на производительность, а размер узла дерева должен быть равен определенному значению, в противном же случае скорость вставок не будет оптимальной. В дальнейшем планируется исследовать влияние других параметров LSM-дерева для получения более подробных результатов.

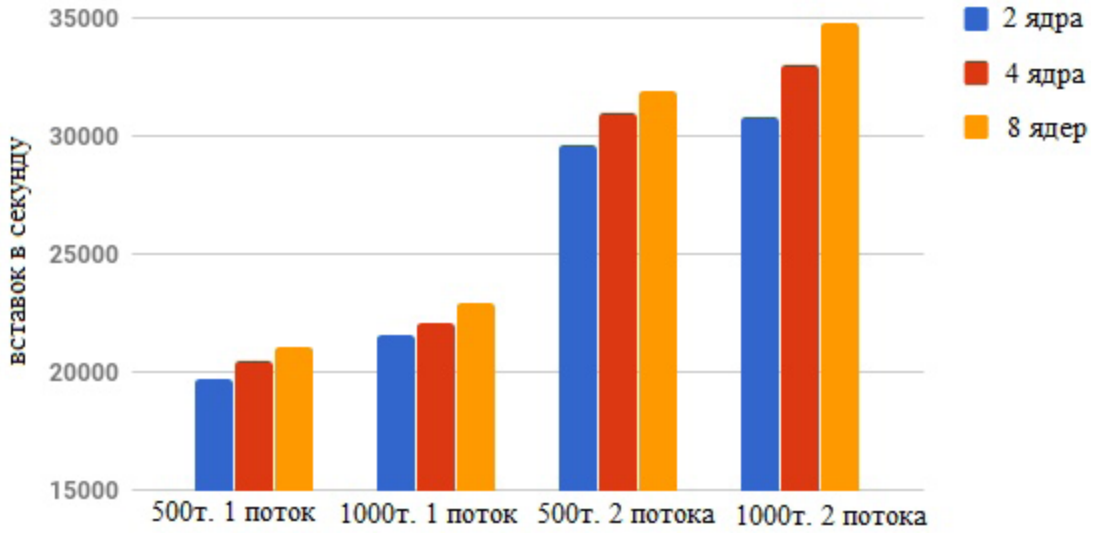


Рис. 5. Влияние распараллеливания на производительность

Литература

1. Log Structured Merge Trees. – Режим доступа: http://www.benstopford.com/2015/02/14/log-structured-merge-trees/

2. The log-structured merge-tree (LSM-tree). UMass/Boston Math & CS Dept Technical Report: O'Нил П.Е. [и др.] – Acta Informatica, 1991. – 32 с.

3. Фильтр Блума. – Режим доступа: http://codeforces.com/blog/entry/5039

4. Cs265 systems project: an LSM-tree based key-value store. – Режим доступа: http://daslab.seas.harvard.edu/classes/cs265/project.html

5. *Аггарвал, А.* The input/output complexity of sorting and related problems / А. Аггарвал, Дж. С. Виттер // Commun. ACM, 31 (9) – 1988. – С.1116-1127.

6. *Грефе, Г.* Modern B-tree Techniques / Гётч Грефе. – 2010. –Vol. 3, No. 4. – 203 с.

Глазырин Игорь Борисович – магистрант 2-го года обучения кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: heathwind@yandex.ru

Борисенков Дмитрий Васильевич (научный руководитель) – канд. техн. наук, доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: xuser@relex.ru

УДК 51-7

РАЗРАБОТКА МАТЕМАТИЧЕСКОЙ МОДЕЛИ ПЕРЕДАЧИ ДАННЫХ В БЕСПРОВОДНЫХ СЕТЯХ

В. Е. Глушаков

Воронежский государственный университет

Тенденция развития информационных технологий требует обработки возрастающего объёма информации. Для этого всё чаще используют распределённые системы обработки информации с использованием беспроводных сетевых технологий. Однако их использование ограничено в настоящее время из-за временных задержек в сети и потери пакетов. Для расчёта параметров функционирования таких систем необходима разработка математических моделей, позволяющих учитывать эти факторы при проектировании систем для оптимизации времени.

В статье разработана математическая модель передачи пакета в сети Wi-Fi между двумя станциями, позволяющая определить возможное время доставки [1].

Данная модель была построена для стандарта 802.11, использующего метод множественного доступа с контролем несущей и предотвращением коллизий (CSMA/CA) [2].

При построении модели была использована последовательность обмена кадрами, представленная на рис. 1 [3] (для стандарта 802.11a).

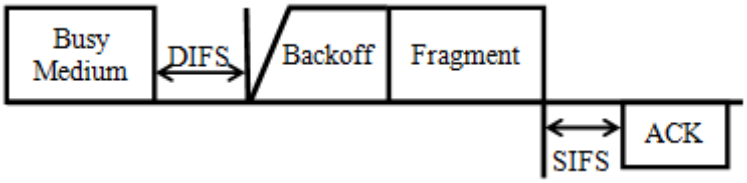


Рис. 1. Временная последовательность обмена пакетом между узлами сети

При отправке пакетов каждое из устройств выполняет следующие действия (рис. 1):

• станция оповещает устройство на другом конце, что готова к отправке, а требуемый адресат – к приему данных (под Busy Medium понимается контроль несущей);

• убедившись, что канал свободен, перед началом передачи данных станция выжидает в течение определенного промежутка времени, который складывается из двух частей: промежутка DIFS и промежутка обратного отсчета Backoff Time (Backoff), выбираемого случайным образом по определенному алгоритму, описанному в [2,12];

• если в течение всего промежутка ожидания канал остается свободным, то узел начинает передачу кадра данных (Fragment);

• между кадром данных (Fragment) и его подтверждением (кадр ACK) должен быть наименьший из межкадровых интервалов SIFS.

Процесс обмена пакетами будем считать однородным Марковским с дискретными состояниями и непрерывным временем. Тогда передаче данных будет соответствовать размеченный граф состояний, представленный на рис. 2.

Система может принимать следующие состояния:

0 – начальное состояние (нет пакетов),

1 – генерация пакета передающей станцией,

2 – пауза 1 (станция ждет время DIFS+ Backoff Time),

3 – передача 1 пакета,

4 – пауза 2 (станция ждет время SIFS),

5 – передача пакета подтверждения ACK принимающей станцией.

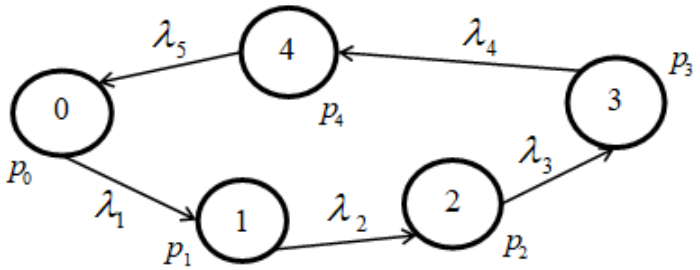


Рис. 2. Граф состояний передающей станции

Так как для двух находящихся в сети станций одновременная передача информации невозможна, появляется новое состояние – пауза (6), во время которой одна из станций ждет, пока другая закончит передачу и освободит канал. В этом случае размеченный граф состояний будет иметь вид, представленный на рис. 3 [1] .

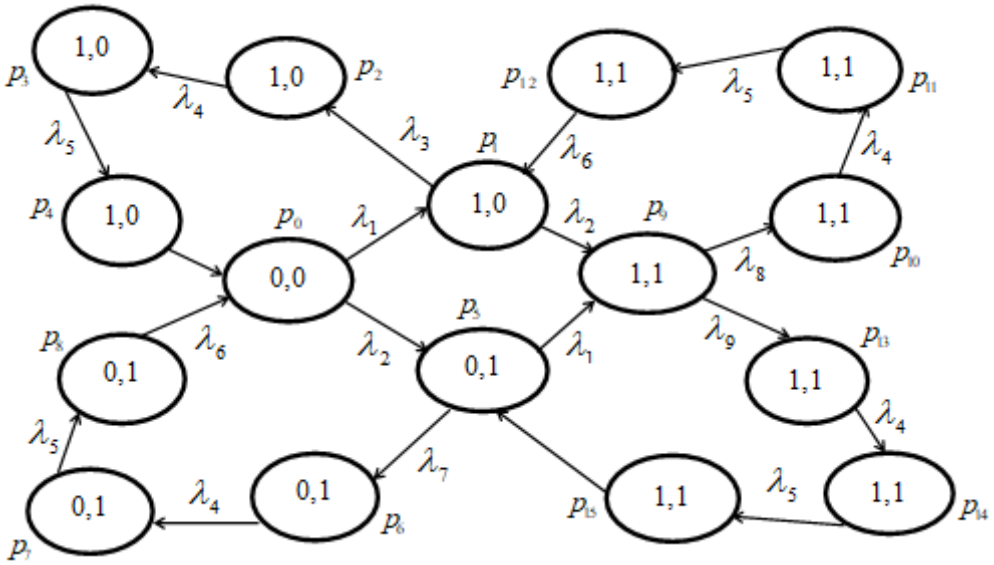


Рис. 3. Граф для 16-ти состояний

Здесь каждое состояние характеризуется двумя параметрами – наличием пакетов для передачи в первом и втором устройствах (0 – пакета для отправки нет, 1 – пакет для отправки есть). Возможные состояния:

- p_0 – пакета для отправки нет ни у первой, ни у второй станций;
- p_1 – пакета для отправки второй станцией нет, а для первой – есть;
- $p_2 - p_4$ – пакета для отправки второй станцией нет, первая станция осуществляет передачу пакета;
- p_5 – пакета для отправки первой станцией нет, а для второй – есть;
- $p_6 - p_8$ – пакета для отправки первой станцией нет, вторая станция осуществляет передачу пакета;
- p_9 – есть пакеты для отправки как первой станцией, так и второй;
- $p_{10} - p_{12}$ – происходит передача одного фрагмента второй станцией, первая станция находится в состоянии «пауза» (она ожидает, пока вторая станция закончит передачу и освободит канал);
- $p_{13} - p_{15}$ – происходит передача одного фрагмента первой станцией, (вторая станция находится в состоянии «пауза».

Здесь λ_j пакетов/с – интенсивность передачи информации j-й станцией $(j = 1,2)$, λ_i – интенсивность передачи информации, $\lambda_i = 1/t_i \quad (i = 3,\ldots,9)$, $t_3 = t_{DIFS} + t_{BACKOFF\_1}$, $t_4 = t_{FRAGMENT}$, $t_5 = t_{SIFS}$, $t_6 = t_{ACK}$, $t_7 = t_{DIFS} + t_{BACKOFF\_2}$, $t_8 = t_{DIFS} + t_{BACKOFF\_3}$, $t_9 = t_{DIFS} + t_{BACKOFF\_4}$.

Объединим состояния 3, 4 в одно и построим новый размеченный граф состояний, представленный на рис. 4.

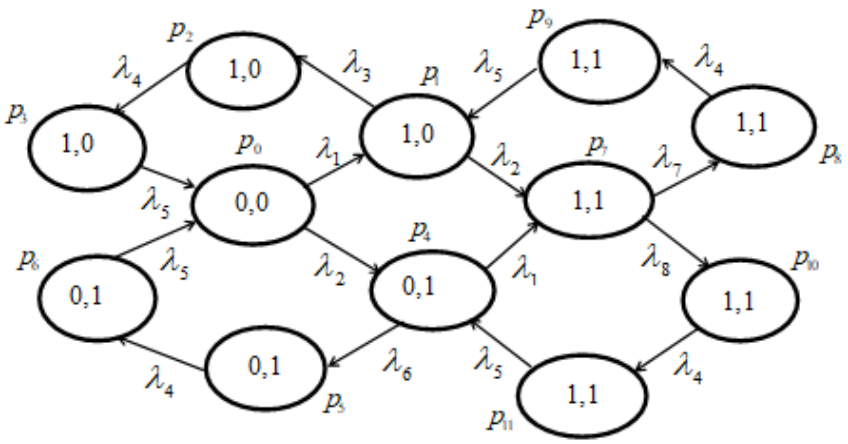


Рис. 4. Граф для 12-ти состояний

Здесь $t_3 = t_{DIFS} + t_{BACKOFF\_1}$, $t_4 = t_{FRAGMENT}$, $t_5 = t_{SIFS} + t_{ACK}$, $t_6 = t_{DIFS} + t_{BACKOFF\_2}$, $t_7 = t_{DIFS} + t_{BACKOFF\_3}$, $t_8 = t_{DIFS} + t_{BACKOFF\_4}$.

После преобразования графа состояния 2, 3–4 опять объединим в одно, получим размеченный граф состояний, представленный на рис. 5. Здесь состояния, в которых завершена доставка пакета от первого устройства, для определения закона распределения сделаны конечными.

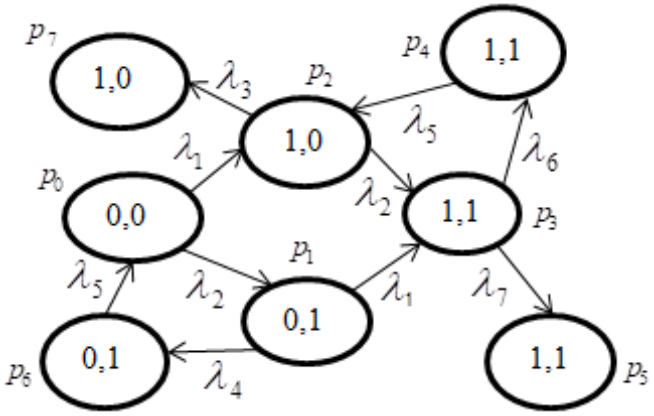


Рис. 5. Граф для 8-ми состояний

Здесь $t_3 = t_{DIFS} + t_{BACKOFF\_1} + t_{FRAGMENT}$, $t_4 = t_{DIFS} + t_{BACKOFF\_2} + t_{FRAGMENT}$, $t_5 = t_{SIFS} + t_{ACK}$, $t_6 = t_{DIFS} + t_{BACKOFF\_3} + t_{FRAGMENT}$, $t_7 = t_{DIFS} + t_{BACKOFF\_4} + t_{FRAGMENT}$.

Для нахождения математического ожидания времени доставки пакета первой станцией надо выписать систему уравнений Колмогорова, предложенную в [1]. Сделаем это для графа, изображенного на рис. 5 (для двух других случаев системы строятся аналогично). Соответствующая система уравнений имеет вид:

$$\begin{cases} \dfrac{dp_0}{dt} = \lambda_5 p_6 - \lambda_1 p_0 - \lambda_2 p_0, \\ \dfrac{dp_1}{dt} = \lambda_2 p_0 - \lambda_1 p_1 - \lambda_4 p_1, \\ \dfrac{dp_2}{dt} = \lambda_1 p_0 + \lambda_5 p_4 - \lambda_3 p_2 - \lambda_2 p_2, \\ \dfrac{dp_3}{dt} = \lambda_1 p_1 + \lambda_2 p_2 - \lambda_6 p_3 - \lambda_7 p_3, \end{cases}$$

$$\begin{cases} \dfrac{dp_4}{dt} = \lambda_6 p_3 - \lambda_5 p_4, \\ \dfrac{dp_5}{dt} = \lambda_3 p_2 + \lambda_7 p_3, \\ \dfrac{dp_6}{dt} = \lambda_4 p_1 - \lambda_5 p_6. \end{cases}$$

Добавим начальные условия:

$$\begin{cases} p_0(0) = 1 \\ p_i(0) = 0 \quad (i = 2, \ldots, 6) \end{cases}.$$

Для численного решения этой системы с заданными начальными условиями использовались значения параметров из стандарта IEEE 802.11a [3, с. 16]: $t_{SLOT\_TIMER} = 9$ *мкс*, $t_{SIFS} = 16$ *мкс*, $t_{DIFS} = t_{SIFS} + 2t_{SLOT\_TIMER} = 34$ *мкс*, $t_{BACKOFF\_i} = d_i \cdot t_{SLOT\_TIMER}$ (d_i – случайная величина таймера отката *Backoff Time*, в данном случае $d_i = 30$ $(i = 1, \ldots, 4)$), размер фрейма *ACK* – 14 байт, размер передаваемого кадра (фрейма) *FRAGMENT* – 798 байт (770 байт плюс 28 байт служебной информации). Скорость передачи данных K – 100 Мбит/с. Решение системы находилось численно на отрезке [0; 0.25] с числом отрезков разбиения $N = 1500$.

Для рассматриваемого случая (рис. 5) плотность распределения вероятностей времени доставки пакетов первой станцией определяется формулой:

$$F(t) = p_2(t)\lambda_3 + p_3(t)\lambda_7.$$

Численный эксперимент был проведен для значений λ_i $(i = 1, 2)$ от 200 до 1000.

На приведённых графиках (рис. 6–8) представлены относительные погрешности вычисления математического ожидания времени доставки пакетов первой станцией для разного числа состояний. Частота отправки пакетов первой станцией варьируется.

Поясним, что понимается под записью «%$n(j-i)$». Например, запись «%200(12 – 8)» означает, что для $\lambda_2 = 200$ находится разность значений между 12-ю и 8-ю состояниями для всех значений λ_1, а затем для каждого из них определяется относительная погрешность вычисления.

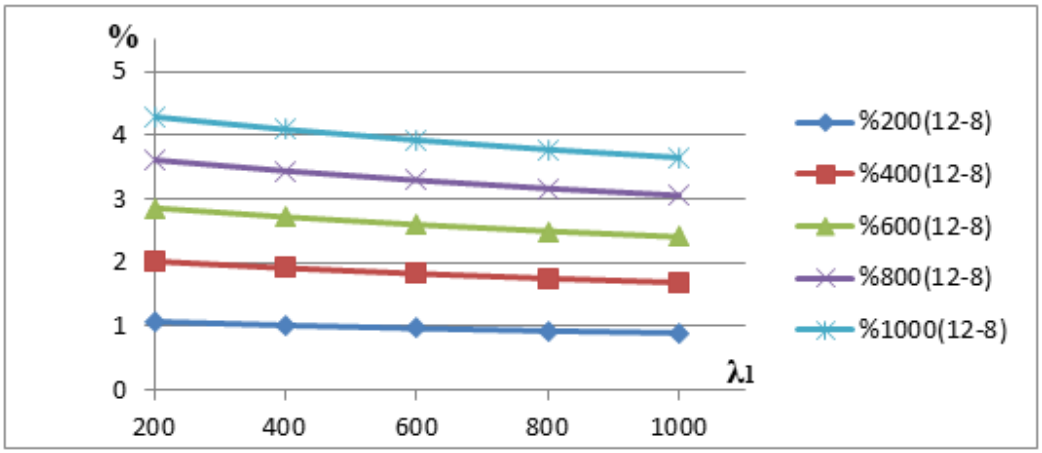


Рис. 6. Относительная погрешность вычисления времени доставки информации для 12-ти и 8-ми состояний

Из графиков на рис. 6–8 видно, что относительная погрешность для рассмотренных значений λ_i $(i = 1, 2)$ для разного числа состояний не превышает 5 %, поэтому при моделировании процесса передачи данных в сетях Wi-Fi можно сокращать число состояний без особого ущерба для математической модели.

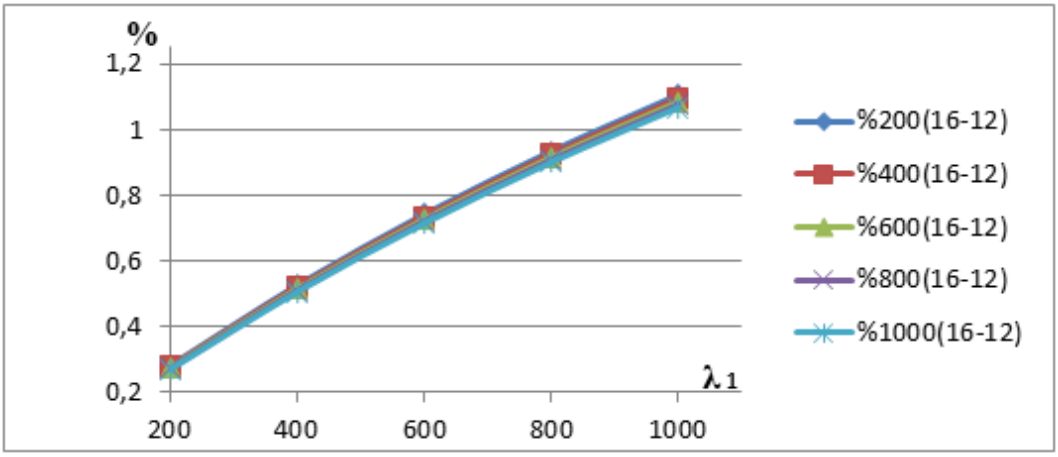


Рис. 7. Относительная погрешность вычисления времени доставки информации для 16-ти и 12-ти состояний

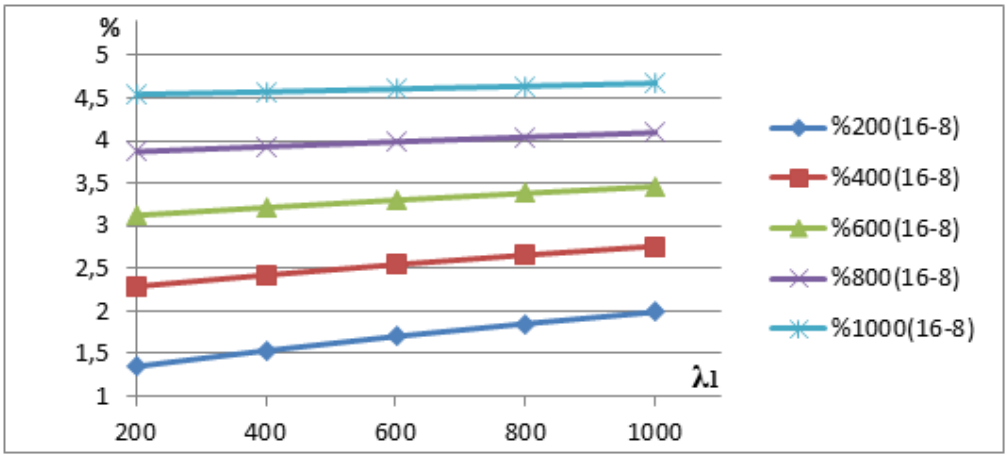


Рис. 8. Относительная погрешность вычисления времени доставки информации для 16-ти и 8-ми состояний

Литература

1. *Глушаков, В. Е.* Исследование подходов к моделированию передачи данных в беспроводных сетях / В. Е. Глушаков // Научный журнал «Globus»: XXX Международная научно-практическая конференция «Достижения и проблемы современной науки», 1 часть. (Санкт-Петербург, 4 мая 2018 г.). – Санкт-Петербург: Научный журнал «Globus», 2018. – С. 48–55. – Режим доступа: https://globus-science.ru/Archive/new/Globus_Multi_May_2018_part_I.pdf.

2. *Платунова, С. М.* Архитектура и технические средства корпоративной сети на базе беспроводного оборудования WI-FI фирмы ZyXEL : учеб. пособие / С. М. Платунова. – Санкт-Петербург : ИТМО, 2014. – 61 с.

3. Раздел 7. Локальные беспроводные сети WiFi. Лекции по стандартам. – Режим доступа: http://docplayer.ru/33818454-Razdel-7-lokalnye-besprovodnye-seti-wifi-lekcii-po-standartam.html (Дата обращения: 3.03.18).

Глушаков Виталий Евгеньевич – аспирант 2-го года обучения Воронежского государственного университета. E-mail: vitalikgl@gmail.com

Абрамов Геннадий Владимирович (научный руководитель) – д-р техн. наук, проф., профессор кафедры Математического обеспечения ЭВМ. E-mail: agwl@yandex.ru

УДК 004.9

РАЗРАБОТКА СИСТЕМЫ МОНИТОРИНГА, АНАЛИЗА И ПРОГНОЗИРОВАНИЯ НЕИСПРАВНОСТЕЙ АВТОМОБИЛЕЙ

М. В. Головай

Воронежский государственный университет

Введение

Автомобильные системы сложны как в аппаратном, так и в программном обеспечении, поэтому их обслуживание является сложной задачей. Стратегия обслуживания, используемая в автомобильной промышленности, как правило, является реактивной, что приводит к сокращению срока службы автомобиля, а также к потере денег. Прогнозируемое техническое обслуживание необходимо на данном этапе для преодоления этих проблем.

Европейская комиссия сообщает, что в течение 20 лет произойдет 50 %-е увеличение количества транспортных средств [1]. В связи с этим потребуются эффективные стратегии по поддержанию работоспособности за счет своевременного технического обслуживания. Подробное обсуждение прогностического обслуживания автомобильной промышленности представлено в работе Притца [1], в которой также обсуждается всесторонний анализ того, как подходы машинного обучения используются в автомобильной промышленности для прогнозирования неисправностей. В связи с возрастающей сложностью транспортных средств, промышленность перешла к автоматизированному анализу данных. В то же время, при минимальных затратах на беспроводную связь и растущей тенденции применения мобильных приложений, стало возможным анализировать бортовые данные для прогнозирования неисправностей. Нарьял и др. [4] использовал Android приложение с бортовой системой диагностики транспортных средств для мониторинга здоровья автомобиля, где водитель был уведомлен в случае каких-либо тревожных состояний.

1. Связанные исследования

Был предложен новый алгоритм под названием «Алгоритм последовательной разработки моделей» [5], в котором предложенный алгоритм учитывает модели из гарантийных данных транспортных средств, а изученные модели преобразуются в экспертную систему, основанную на правилах. Затем с развитием автомобилестроения тенденция перешла от диагностики транспортных средств к прогнозированию с использованием сенсорных данных, так как в автомобилях внедрялось большое количество сенсоров. На основе данных этих датчиков была представлена новая модель прогнозирования неисправностей «Консенсусная самоорганизованная модель для прогнозирования неисправностей» Байттнера и др. [2], в которой модель узнала интересные взаимосвязи между данными датчиков в транспортных средствах, а также в малых мехатронных системах. В работе [3] были представлены другие методики изучения классификаторов на основе несбалансированных данных. В 2013 году [6] был представлен комплексный анализ прогнозирования отказов компрессора. Были использованы данные сенсоров, а также подробно обсуждены проблемы и задачи. Данные были собраны из зарегистрированной базы данных транспортных средств, которая ведет учет технического обслуживания всех транспортных средств, посещающих СТО Volvo. Для прогнозирования неисправностей компрессора были использованы Random Forest, KNN и C5.0. Система контролирует

активность водителя и состояние двигателя автомобиля. Sun представила мультисенсорную методику синтеза [7] для мониторинга состояния автомобиля с использованием данных о масле и вибрационных сигналов. Шмальцель и др. представили всеобъемлющую дискуссию о том, как можно использовать интеллектуальные датчики для мониторинга состояния системы и диагностики проблемы [8].

2. Предлагаемое решение

Для каждого из вышеперечисленных подходов существует множество вариантов диагностического и прогностического обслуживания. Предлагается построить полную инфраструктуру системы удаленного мониторинга состояния транспортного средства и прогностического обслуживания с использованием методологии, основанной на данных, собранных в режиме реального времени, когда транспортное средство находится в движении. Для генерации данных планируется использовать Hyundai Solaris 2014 года выпуска. Удаленный мониторинг состояния относится к мониторингу работы различных систем автомобилей дистанционно и прогнозирование относится к прогнозированию неисправностей заранее. Целью исследования является прогнозирование неисправностей в четырех основных системах автомобиля и обеспечение в режиме реального времени мониторинга транспортных средств и прогностического обслуживания системы.

Архитектура имеет три основных слоя, как показано на рис. 1. На первом уровне генерируются данные. Сканер OBD2 подключается к автомобилю через порт OBD2. На рынке представлено множество таких сканеров; в предлагаемой системе, которая по сути является микроконтроллером, используется ELM 327 Bluetooth (ELM 327). Этот сканер ведет себя как мост между транспортным средством и портативным устройством, то есть мобильным или ноутбуком, который поддерживает Bluetooth. Существуют крошечные ИС (интегральные схемы), генерируемые для этой связи между транспортным средством и портативным устройством. Все данные датчика генерируются, когда транспортное средство находится в движении и непрерывно передаются на смартфон через Bluetooth. На уровне обработки данных первым шагом является выбор показателей на основе предложения экспертов. атем на массиве данных применяется PCA (Принципиальный компонентный анализ) для уменьшения признаков. После этого на этапе классификации используются четыре алгоритма классификации, включая дерево решений, случайный лес, K-NN и SVM. Изучаются интересные комбинации данных или отношений, и дальнейшая обработка выполняется на стороне сервера. Полученные результаты сохраняются на сервере для последующего вывода, который используется для прогнозирования неисправностей и удаленного мониторинга транспортного средства. На уровне удаленного мониторинга владелец или заинтересованное лицо транспортного средства может удаленно следить за текущим состоянием транспортного средства, например, за состоянием топлива, скоростью и текущим положением. Водитель или владелец транспортного средства уведомляется о выходе из строя любой подсистемы транспортного средства посредством автоматического оповещения.

Заключение

Основными преимуществами предложенной системы являются следующие:

1. Неквалифицированный человек может вникнуть в сбой любой подсистемы транспортного средства, поскольку решение предоставляет всю информацию о состоянии и статусе транспортного средства на смартфоне, который подключен к транспортному средству.

2. Система ориентирована на реальные решения, использующие методы машинного обучения, которые также могут сэкономить деньги и время.

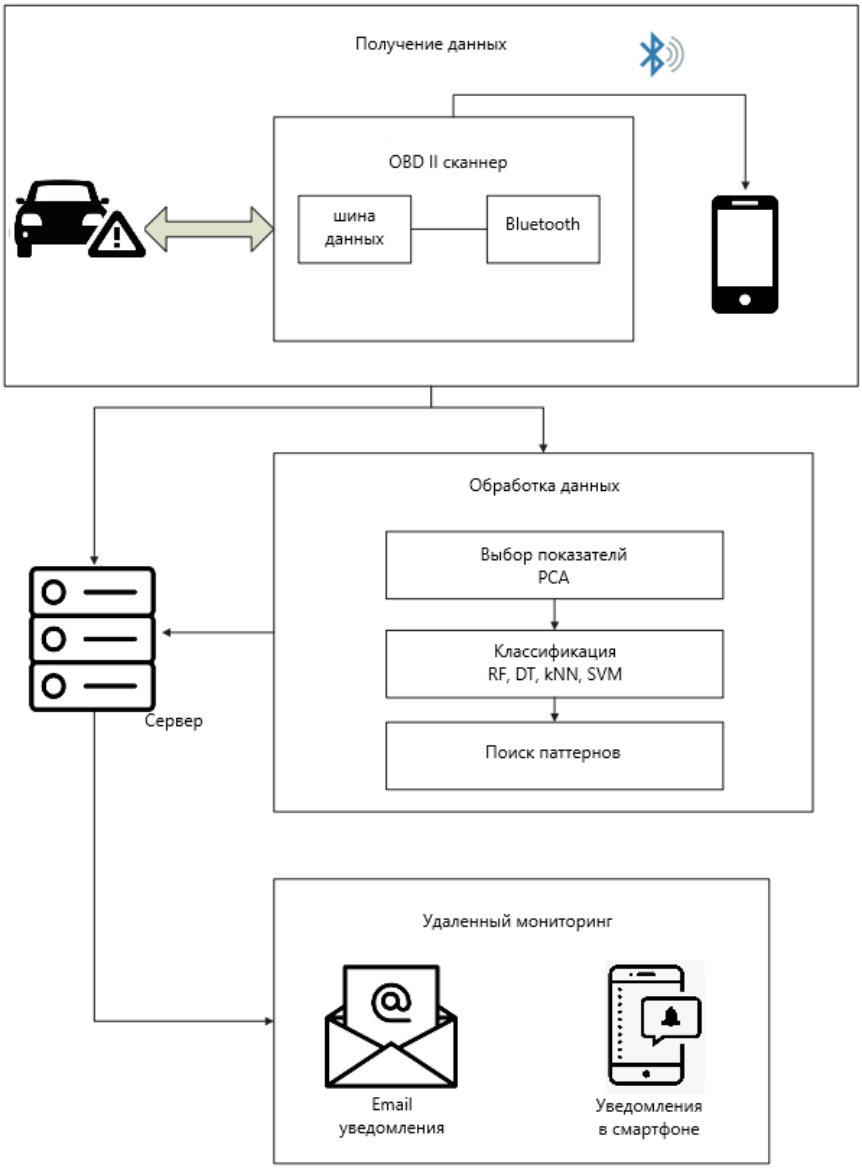


Рис. 1.

3. Срок службы транспортного средства увеличивается, поскольку владелец/водитель знает все о внутреннем состоянии систем; затем он может сделать несколько шагов, чтобы избавиться от любого системного сбоя.

4. Риск несчастного случая снижается с учетом постоянной информированности о состоянии автомобиля.

Благодарности

Выражаю огромную благодарность моему научному руководителю – Сафронову В. В. за неизмеримую поддержку и веру в мои силы, а так же Кениной Н. А. за мотивацию к выполнению данной работы и написанию статьи.

Литература

1. *Prytz, R.* Machine Learning Methods for Vehicle Predictive Maintenance Using Of-Board And On-Board Data: диссертация – Halmstad: Halmstad University, 2014. – 20 с.

2. Consensus self-organized models for fault detection (COSMO) / S. Byttner [et al.] // Engineering Applications of Artifcial Intelligence. – 2011. – No 24. – P. 833–839.

3. Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data / R. Prytz [и др.] // Engineering Applications of Artifcial Intelligence. – 2015. – No 41. – P. 139–150.

4. *Naryal, E.* Real time vehicle health monitoring and driver information display system based on CAN and Android / E. Naryal, P. Kasliwal // International Journal of Advance Foundation and Research in Computer. – 2014. – №1. – С. 76–84.

5. *Buddhakulsomsiri, J.* Sequential pattern mining algorithm for automotive warranty data / J. Buddhakulsomsiri, A. Zakarian // Computers & Industrial Engineering. – 2009. – No 57. – P. 137–147.

6. *Stefanowski, J.* Overlapping, rare examples and class decomposition in learning classifers from imbalanced data / J. Stefanowski // Smart Innovation, Systems and Technologies. – 2013. – No 13. – P. 277–306.

7. *Sun, Q.* Sensor fusion for vehicle health monitoring and degradation detection / Q. Sun // Proceedings of the 5th International Conference on Information Fusion (США, *Июль* 2002 г.), – США, 2002. – С. 1422–1427.

8. *Schmalzel, J.* An architecture for intelligent systems based on smart sensors / J. Schmalzel [et al] // IEEE Transactions on Instrumentation and Measurement. – 2005. – No 54. – P. 1612–1616.

Г**оловай Михаил Витальевич** – магистрант 1-го года обучения кафедры ERP систем и бизнес процессов Воронежского государственного университета. E-mail: mikhail.golovai@gmail.com

Сафронов Виталий Владимирович (научный руководитель) – канд. техн. наук, проф., доцент кафедры ERP систем и бизнес процессов Воронежского государственного университета. E-mail: vitolik@bk.ru

УДК УДК 004.9

РАЗРАБОТКА СИМУЛЯТОРА РОБОТА PTC P300 НА ИГРОВОМ ДВИЖКЕ UNITY

Н. Г. Горбунов

Воронежский государственный университет

Введение

Почти во всех приложения есть анимация, но для того, чтобы привести в движение те или иные части модели, необходимо использовать алгоритмы скелетной анимации [3]. Симуляторы можно использовать не только в игровой индустрии, но и в промышленной. Если рассматривать промышленность, то здесь симуляторы необходимы для обучения персонала: производство промышленных объектов является трудоемкой работой, а сами объекты в конечном итоге является дорогостоящим объектом, в связи с чем обучать персонал на реальной машине практически невозможно. Именно поэтому нужен симулятор – максимально погрузиться в работу и управление объектом и при этом не бояться ошибиться или что-то сломать.

Постановка задачи

Необходимо создать симулятор робота (рис. 1) на игровом движке Unity [2] для обучения персонала. Робот состоит из пяти основных частей: платформа, на которой расположены все части робота, гусеницы, четыре лапы, которые отвечают за устойчивость робота во время его работы, основание, на котором закреплена стрела, и сама стрела, состоящая из пяти последовательно соединенных сегментов. Все перечисленные части должны двигаться согласованно. Более того, сами части так же состоят из более мелких деталей, которые отвечают за целостность конструкции.

Необходимо реализовать такие операции как:

1. Движение робота.
2. Поворот робота.
3. Подъем и опускание лап с фиксацией их на земле.
4. Поворот каждого сегмента стрелы.

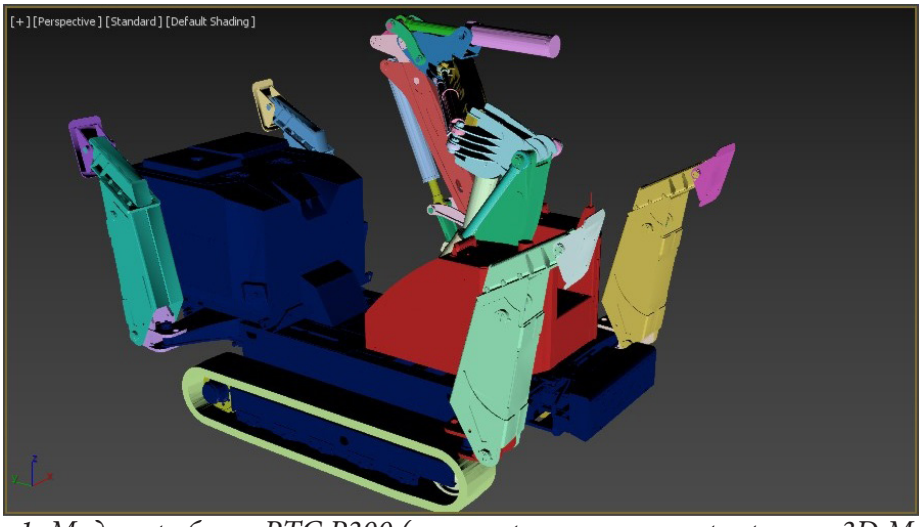


Рис. 1. Модель робота PTC P300 (окно перспективы в программе 3D MAX)

1. Передвижение робота

Поворот робота (рис. 1), как и его перемещение вперед и назад, осуществляется относительно центра объекта.

Общая идея реализации перемещения: при перемещении вперед или назад выбирается направляющий вектор, относительно которого происходит движение, а также число, на которое происходит перемещение объекта. В Unity данное значение отвечает за скорость перемещения, то есть чем выше или ниже данное значение, тем быстрее или медленнее будет перемещаться в пространстве объект.

2. Повороты

При повороте у объекта выбирается точка, относительно которой происходит поворот, направляющий вектор, относительно которого происходит поворот, а также значение угла в радианах.

В среде разработки Unity повороты вычисляются как скорость поворота (коэффициент поворота), умноженный на 1 градус [2]. Поворот можно производить двумя способами: либо оставить направляющий вектор одинаково направленным при повороте как влево, так и вправо, а коэффициент поворота брать с разными знаками, либо коэффициент поворота остается неизменным, но меняется направляющий вектор, относительно которого происходит поворот.

2.1. Поворот основания

Основание (рис. 2) вращается аналогично тому, как поворачивается весь робот, то математика поворота используется аналогичная. Только теперь поворот осуществляется не всего робота целиком, а только его основания и всех связных с ним деталей.

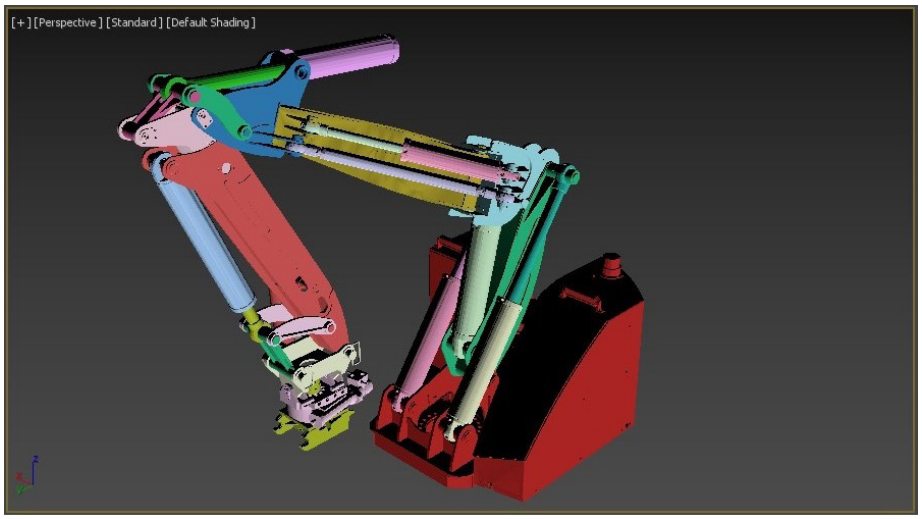


Рис. 2. Основание стрелы и связные с ней детали (окно перспективы в программе 3D MAX)

2.2. Поворот первого сегмента стрелы

Все повороты осуществляются за счет пересчета изменения углов между деталями стрелы, то необходимо посчитать изменения углов между деталями, которые в момент поворота не находятся в состоянии покоя. Рассмотрим изменения углов на примере первого сегмента стрелы (рис. 3).

Отрезок OB – деталь первого сегмента стрелы, которая является основной, так как соединяет другую часть стрелы с основанием. Система «поршень+цилиндр» лежит на отрезке AB и с физической точки зрения отвечает за поворот стрелы (а точнее наклон ее вперед и назад).

В начальный момент времени рассчитаем длины векторов $\overline{OA}$ и $\overline{OB}$ по формуле (1), которые являются постоянными. Так как стрела и поршень лежат в параллельных плоскостях, координатой x можно пренебречь.

$$\rho = \sqrt{(y_1 - y_2)^2 + (z_1 - z_2)^2}. \tag{1}$$

Длина отрезка AB будет изменяться в процессе движения. В начальном состоянии его длина так же может быть рассчитана по формуле (1).

Введем обозначения: пусть $l = \left|\overline{OA}\right|$, $d = \left|\overline{OB}\right|$, $h = \left|\overline{AB}\right|$. Для того чтобы вычислить угол $\alpha = \angle AOB$ в начальном состоянии, нужно воспользоваться теоремой косинусов:

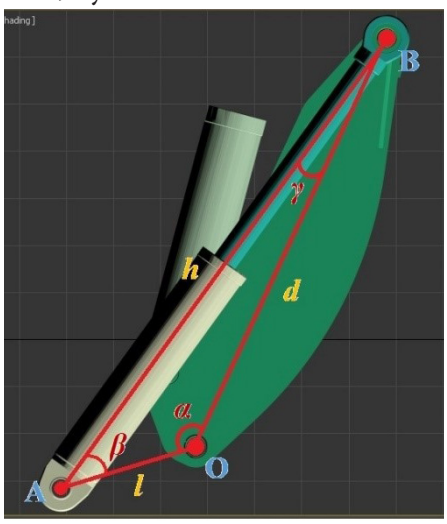


Рис. 3. Углы, влияющие на поворот первого сегмента стрелы

$$\cos\alpha = \frac{l^2 + d^2 - h^2}{2ld} \tag{2}$$

Исходя из физических ограничений и на данном этапе пренебрегая другими деталями, которые могут влиять на ход движения первого сегмента, получим, что $\alpha \in (0,180)$. Функция $y = \arccos x$ определена на отрезке $x \in [-1;1]$ и принимает значение $y \in [0,180]$. Поэтому можно сделать вывод, что α точно определяется по следующей формуле:

$$\alpha = \arccos\frac{l^2 + d^2 - h^2}{2ld} \tag{3}$$

Определим теперь угол $\beta = \angle OAB$. Для этого сначала рассмотрим теорему синусов:

$$\frac{\sin\beta}{d} = \frac{\sin\alpha}{h},$$

$$\sin\beta = \frac{d}{h}\sin\alpha. \tag{4}$$

Затем по теореме косинусов получаем:

$$\cos\beta = \frac{l^2 + h^2 - d^2}{2lh} \tag{5}$$

Зная синус и косинус угла β, вычисленные по формулам (4) и (5) соответственно, можно точно определить его величину.

Угол $\gamma = \angle OBA$ определим из суммы углов треугольника:

$$\alpha + \beta + \gamma = 180,$$
$$\gamma = 180 - \alpha - \beta. \quad (6)$$

Таким образом, получаем полностью определенный треугольник $\triangle OAB$.

Теперь рассмотрим процесс изменения угла при повороте.

Пусть происходит поворот стрелы. Будем рассматривать его как изменение угла между исходным положением стрелы и его текущим положением.

Пусть в следующий момент времени получаем угол $\alpha' = \alpha + \Delta\alpha$, то есть стрела поворачивается на угол $\Delta\alpha$ относительно предыдущего положения. По теореме косинусов из формулы (2) получаем длину сжатого или растянутого отрезка AB:

$$h' = |AB| = \sqrt{l^2 + d^2 - 2ld\cos\alpha}.$$

По формулам (4) и (5) получаем новое значение угла $\angle OAB$:

$$\beta = f(\cos\beta', \sin\beta').$$

Таким образом цилиндр поворачивается на угол $\Delta\beta = \beta' - \beta$ относительно своего исходного положения.

Новый угол $\angle OBA$ получим по формуле (6), поэтому поршень поворачивается на угол $\Delta\gamma = \gamma' - \gamma$ относительно своего исходного положения.

В итоге, получаем величины поворотов соответствующих объектов первого сегмента стрелы и значения новых углов, используемых для дальнейших расчетов.

Так как физика поворота первого сегмента стрелы и прилежащих деталей имеет сложную логику со своими нюансами, то стоит рассмотреть алгоритм поворота данного сегмента подробнее.

Алгоритм поворота первого сегмента:

1. Вычисляется $\Delta\alpha$ как произведение скорости по ворота на единицу угла;

2. Поворачиваем отрезок $\overline{OB}$, а вместе с ним и всю стрелу, на угол $\Delta\alpha$. При этом углы β и γ остаются неизменными.

3. Вычисляется расстоянием между точками A и B по формуле:

$$h = |AB| = \sqrt{l^2 + d^2 - 2ld\cos\alpha}.$$

4. Вычисляется угол β' как

$$\beta' = \arctan\frac{\sin\beta'}{\cos\beta'}, \text{ где}$$
$$\sin\beta' = \frac{d}{h}\sin\alpha$$
$$\cos\beta' = \frac{l^2 + h^2 - d^2}{2lh}.$$

5. Вычисляется $\Delta\beta$ как разница между значением нового угла β', который появился в результате поворота стрелы на угол $\Delta\alpha$, и значением угла β, которое было до поворота.

6. Поворачиваем цилиндр (лежит на отрезке $\overline{AB}$) на угол $-\Delta\beta$.

7. Вычисляется угол γ' из суммы углов треугольника.

8. Вычисляется $\Delta\gamma$ как разница между значением нового угла γ', который появился в результате поворота стрелы на угол $\Delta\alpha$, и значением угла γ, которое было до поворота.

9. Поворачиваем поршень (лежит на отрезке $\overline{AB}$), входящий в стрелу, на угол $\Delta\gamma$.

Нюанс заключается именно в пунктах 8 и 9, потому что при повороте стрелы на угол $\Delta\alpha$ необходимо доворачивать цилиндр и поршень на соответствующие углы, но в противополож-

ные относительно друг друга стороны, для того чтобы поршень осуществлял вход в цилиндр без всяких препятствий и не ломал систему, которую образует отрезок $\overline{AB}$ (рис. 7).

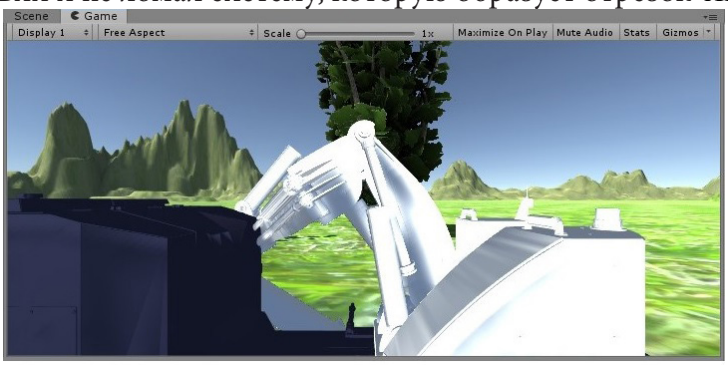


Рис. 4. Поломка системы поршень-цилиндр из-за неправильно посчитанных углов

2.2. Поворот четвертого сегмента стрелы

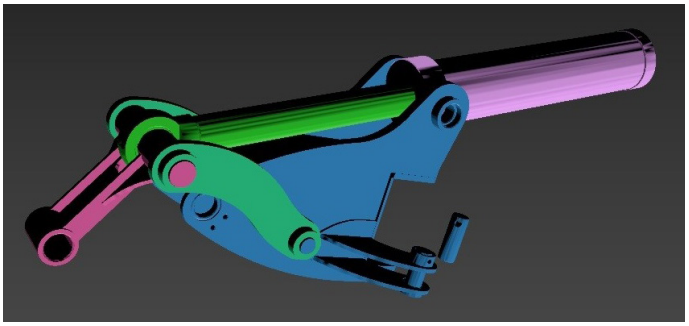


Рис. 5. Четвертый сегмент стрелы

Четвертый сегмент стрелы (рис. 5) отличается от выше перечисленных сегментов тем, что работа углов завязана не на трех точках, а пяти, т. е. в данном случае необходимо пересчитывать больше углов, которые влияют на работу как данного сегмента, так и всей стрелы в целом.

Если рассматривать треугольник $\triangle OAB$, то вычисления углов α, β и γ происходят аналогичным образом, что и вычисления углов в первом, втором и третьем сегментах. Нюанс заключается в работе четырехугольника $BOCD$.

В начальный момент времени, когда четвертый сегмент находится в исходном положении, известны все необходимые для работы сегмента величины. Пусть $h_2 = \left|\overline{BC}\right|$, которое можно вычислить по формуле (1). По теореме косинусов находим угол $\theta = \angle BDC$ в начальном состоянии:

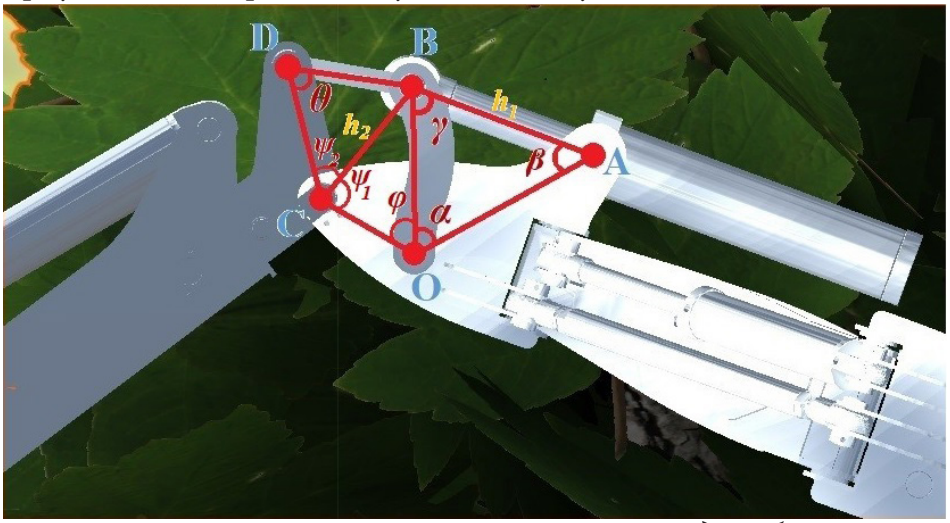


Рис. 6. Четвертый сегмент стрелы в раскрытом виде с обозначениями

$$\theta = \frac{BD^2 + CD^2 - h_2{}^2}{2\,BD\,CD}. \tag{7}$$

Угол $\angle OCD$ делится отрезком h_2 на два угла: ψ_1 и ψ_2. Угол ψ_1 также можно определить по теореме косинусов:

$$\psi_1 = \frac{OC^2 + h_2{}^2 - CB^2}{2\,OC\,h_2}. \tag{8}$$

Используя теоремы синусов и косинусов, можно вычислить угол ψ_2:

$$\frac{\sin\psi_2}{BD} = \frac{\sin\theta}{h_2}$$

$$\sin\psi_2 = \frac{BD}{h_2}\sin\theta. \tag{9}$$

Затем по теореме косинусов получаем:

$$\cos\psi_2 = \frac{CD^2 + h_2{}^2 - BD^2}{2\,CD\,h_2} \tag{10}$$

Зная синус и косинус угла ψ_2, вычисленные по формулам (9) и (10) соответственно, можно точно определить его величину.

Аналогичным способом определяется угол φ:

$$\frac{\sin\varphi}{h_2} = \frac{\sin\psi_1}{OB},$$

$$\sin\varphi = \frac{h_2}{OB}\sin\psi_1. \tag{11}$$

Затем по теореме косинусов получаем:

$$\cos\varphi = \frac{OC^2 + OB^2 - h_2{}^2}{2\,CD\,OB}. \tag{12}$$

Зная синус и косинус угла φ, вычисленные по формулам (11) и (12) соответственно, можно точно определить его величину.

Таким образом, получаем полностью определенный четырехугольник $BOCD$.

Пусть происходит поворот стрелы. Будем его рассматривать как изменение угла между стрелами, что в свою очередь повлечет изменение на ту же величину.

Пусть в следующий момент времени получаем угол $\varphi' = \varphi + \Delta\varphi$, то есть детали, которые привязаны к четвертому сегменту стрелы через точки B, O, C, D поворачиваются на угол $\Delta\varphi$ относительно предыдущего положения. По теореме косинусов получаем расстояние между точками B и C:

$$h'_2 = |BC| = \sqrt{OC^2 + OB^2 - 2\,OC\,OB\cos\varphi'}.$$

Одновременно с этим по теореме косинусов получаем углы ψ_1 и ψ_2:

$$\cos\psi_1 = \frac{OC^2 + h_2{}^2 - OB^2}{2\,OC\,h_2}$$

$$\cos\psi_2 = \frac{CD^2 + h_2{}^2 - BD^2}{2\,CD\,h_2}$$

Несмотря на то, что вычисляем углы по отдельности, нас интересует изменение общего угла $\psi = \psi_1 + \psi_2$, которое можно вычислить по формуле:

$$\Delta\psi = |\psi'_1 - \psi_1| + |\psi'_2 - \psi_2|. \tag{13}$$

Таким образом, детали, которые привязаны к четвертому сегменту в точке C, будут поворачиваться на угол $\Delta\psi$ относительно своего исходного положения.

Новый угол $\theta = \angle BDC$ получаем по формуле (7), поэтому отрезок CD в точке D будет поворачиваться на угол $\Delta\theta = \theta' - \theta$. Причем, как и в случае с изломом, чтобы конструкция четвертого сегмента не сломалась, при уменьшении угла ψ необходимо увеличивать угол θ, поэтому поворот будет осуществляться на $-\Delta\theta$.

Заключение

При выполнении поставленной задачи был изучен игровой движок Unity, научились работать с моделью, а так же смогли применить к этой модели алгоритмы скелетной анимации, чтобы применить такие действия к объекту, как его перемещение, поворот, а так же повороты основных частей объекта.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Понарин, Я. П.* Элементарная геометрия (2 т.) / Я. П. Понарин. — М.: МЦНМО, 2004. – С. 84–85 .

2. Руководство Unity. – Режим доступа: https://docs.unity3d.com/ru/530/Manual/UnityManual.html. – (Дата обращения: 20.04.2020).

3. *Сучкова, М. М.* Разработка модели и алгоритмов скелетной анимации для промышленного манипулятора / Сучкова М. М., Горбунов Н. Г., Медведев С. Н. // Актуальные проблемы прикладной математики, информатики и механики: сб. тр. междунар. конф., Воронеж, 11–13 ноября 2019 г. – С. 507–513.

Горбунов Никита Геннадьевич – магистрант 2-го курса магистратуры кафедры Вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: nikort7@yandex.ru

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 004.93

РЕАЛИЗАЦИЯ КЛАССИФИКАТОРА С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ НЕЧЕТКОГО МОДЕЛИРОВАНИЯ

И. В. Данилова

Воронежский государственный университет

Введение

Работа менеджера по работе с персоналом зачастую связана с анализом большого объема данных и последующим принятием решений. Одним из наиболее ответственных решений является определение соответствующего положения в должностной иерархии компании того или иного сотрудника, что можно представить как задачу классификации. Для решения данной задачи может быть использована экспертная система на основе нечеткой логики.

1. Формализация задачи

Задача разрабатываемой системы заключается в том, чтобы на основе числовой (бальной) оценки квалификации сотрудника определить для него соответствующее положение в проекте (роль/должность). В такой системе экспертом определяется перечень квалификационных требований для каждой должности и роли и степень соответствия им.

Модель экспертной системы на основе нечеткой логики представляет собой набор продукционных правил, написанных на естественном языке и использующих входные данные предметной области как основу для принятия решений.

Пусть $qualification\_req_i$ – i-е квалификационное требование, q_{ij} – j-е значение лингвистической переменной «соответствие квалификационному требованию», rnp_i – i-я роль и должность. Таким образом, база правил заполняется типовыми правилами вида:

$$R_i : IF((qualification\_req_1 = q_{ij1})AND(qualification\_req_2 = q_{ij2})AND...$$

$$AND(qualification\_req_n = q_{ijn})) \Rightarrow THEN(role\_and\_position = rnp_i)$$

Реализуется аппроксимация CLR_A-классификатора в виде НПБЗ

$CLR_A\left(o_i, R_j\right) = \begin{cases} 0, \text{ если } o_i \notin R_j \\ 1, \text{ если } o_i \in R_j \end{cases}$; $O = \left\{o_1, ...o_{|o|}\right\}$ – множество классифицируемых объектов; $R = \left\{R_1, ...R_{|R|}\right\}$ – множество классов (правил).

Каждый объект o_i характеризуется значениями типового набора признаков для каждого квалификационного требования $P = \left\{p_1, ...p_{|p|}\right\}$.

В качестве множества классов выступают наборы ролей и должностей, составленных согласно руководству PMbok [3] и представленных в табл. 1.

Для всех квалификационных требований использовалась единая лингвистическая шкала, построенная на основе лингвистической переменной «Оценка квалификационного требования» с универсальным множеством $U = [0;100]$ и множеством термов $\{bad, fair, good, great, excellent\}$. Каждый терм – нечеткое треугольное число (рис. 1).

Выходные лингвистические переменные (связка роль-должность и отрицательный результат “No”) представлены синглтонами – одноточечными нечеткими множествами, функция принадлежности которых равна единице только в одной точке и нулю во всех остальных:

Таблица 1

Перечень ролей и должностей

Роли	Должности
Java Developer	Junior
Python Developer	Middle
System Administrator	Senjor
Web Developer	
Project Manager	
QA	
DB Engineer	

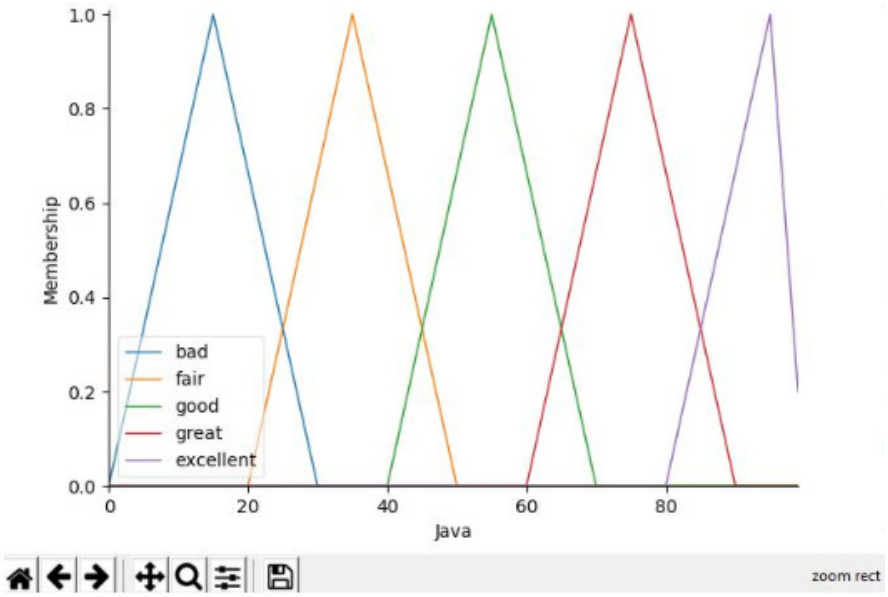


Рис. 1. Лингвистическая шкала лингвистической переменной «Оценка квалификационного требования» (на примере квалификационного требования Java)

$\begin{cases} x = 0, y = 1 \\ x \neq 0, y = 0 \end{cases}$ – для отрицательного результата “No”;

$\begin{cases} x = 50, y = 1 \\ x \neq 50, y = 0 \end{cases}$ – для связки роль-должность.

Дефаззификация выходного значения в этом случае производится методом центра тяжести для синглтонов:

$$U = \frac{\sum_{i=1}^{p} [u_i \mu_i]}{\sum_{i=1}^{p} [\mu_i]},$$

где

U – результат дефаззификации;

u – выходная переменная;

p – количество синглтонов;

μ – функция принадлежности накопленных нечетких множеств;

i – индекс.

2. Реализация и полученные результаты

В качестве технологии для реализации ЭС использовался язык программирования Python, библиотека skfuzzy [4], содержащая API для работы с основными элементами нечеткой логикой.

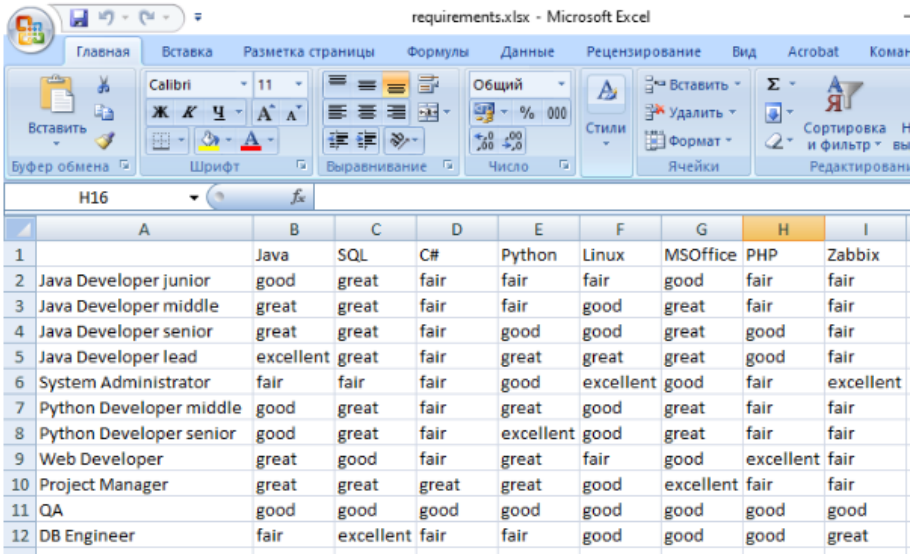
Итоговое правило из программно сформированной нечеткой продукционной базы правил R выглядело следующим образом (на примере правила для класса Java Developer junior):

IF Java[good] AND
SQL[great] AND
C#[fair] AND
Python[fair] AND
Linux[fair] AND
MSOffice[good] AND
PHP[fair] AND
Zabbix[fair]
THEN response[Java Developer junior].

Данная база правил R и является реализацией классификатора CLR_A.

Эксперимент проводился на ПЭВМ со следующими основными параметрами: 2-х ядерный 64-разрядный процессор Intel(R) Pentium(R) 3556U 1.70GHz, 4ГБ ОЗУ.

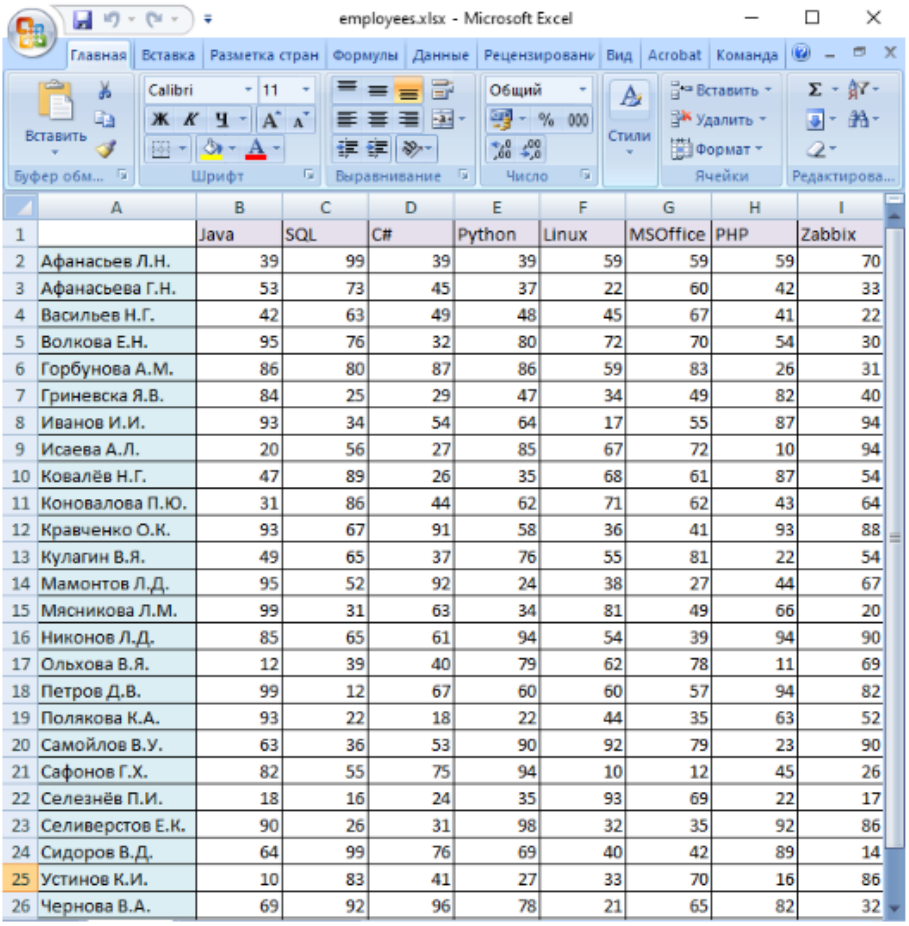
На вход программе поступили числовые параметры для треугольных функций принадлежности терм-множества $\{bad, fair, good, great, excellent\}$ лингвистической переменной «Соответствие квалификационному требованию» и два excel-файла: файл с данными, содержащими квалификационные требования для каждой роли и должности, на основе которых строится база правил, и файл с данными сотрудников. Фрагменты данных файлов представлены на рис. 2 и рис. 3.



	A	B	C	D	E	F	G	H	I
1		Java	SQL	C#	Python	Linux	MSOffice	PHP	Zabbix
2	Java Developer junior	good	great	fair	fair	fair	good	fair	fair
3	Java Developer middle	great	great	fair	fair	good	great	fair	fair
4	Java Developer senior	great	great	fair	good	good	great	good	fair
5	Java Developer lead	excellent	great	fair	great	great	great	good	fair
6	System Administrator	fair	fair	fair	good	excellent	good	fair	excellent
7	Python Developer middle	good	great	fair	great	good	great	fair	fair
8	Python Developer senior	good	great	fair	excellent	good	great	fair	fair
9	Web Developer	great	good	fair	great	fair	good	excellent	fair
10	Project Manager	great	great	great	great	good	excellent	fair	fair
11	QA	good	good	good	good	good	good	good	good
12	DB Engineer	fair	excellent	fair	fair	good	good	good	great

Рис. 2. Требуемое соответствие квалификационному требованию

В результате программа вывела данные по каждому сотруднику и классифицированные должность и роль в компании, которым соответствуют результаты его оценивания менеджером по работе с персоналом. Пример результата работы программы представлен на рис. 4. Время ответа на запрос – 08.306 сек.



	A	B	C	D	E	F	G	H	I
1		Java	SQL	C#	Python	Linux	MSOffice	PHP	Zabbix
2	Афанасьев Л.Н.	39	99	39	39	59	59	59	70
3	Афанасьева Г.Н.	53	73	45	37	22	60	42	33
4	Васильев Н.Г.	42	63	49	48	45	67	41	22
5	Волкова Е.Н.	95	76	32	80	72	70	54	30
6	Горбунова А.М.	86	80	87	86	59	83	26	31
7	Гриневска Я.В.	84	25	29	47	34	49	82	40
8	Иванов И.И.	93	34	54	64	17	55	87	94
9	Исаева А.Л.	20	56	27	85	67	72	10	94
10	Ковалёв Н.Г.	47	89	26	35	68	61	87	54
11	Коновалова П.Ю.	31	86	44	62	71	62	43	64
12	Кравченко О.К.	93	67	91	58	36	41	93	88
13	Кулагин В.Я.	49	65	37	76	55	81	22	54
14	Мамонтов Л.Д.	95	52	92	24	38	27	44	67
15	Мясникова Л.М.	99	31	63	34	81	49	66	20
16	Никонов Л.Д.	85	65	61	94	54	39	94	90
17	Ольхова В.Я.	12	39	40	79	62	78	11	69
18	Петров Д.В.	99	12	67	60	60	57	94	82
19	Полякова К.А.	93	22	18	22	44	35	63	52
20	Самойлов В.У.	63	36	53	90	92	79	23	90
21	Сафонов Г.Х.	82	55	75	94	10	12	45	26
22	Селезнёв П.И.	18	16	24	35	93	69	22	17
23	Селиверстов Е.К.	90	26	31	98	32	35	92	86
24	Сидоров В.Д.	64	99	76	69	40	42	89	14
25	Устинов К.И.	10	83	41	27	33	70	16	86
26	Чернова В.А.	69	92	96	78	21	65	82	32

Рис. 3. Фактическая оценка квалификационных требований персонала Компании

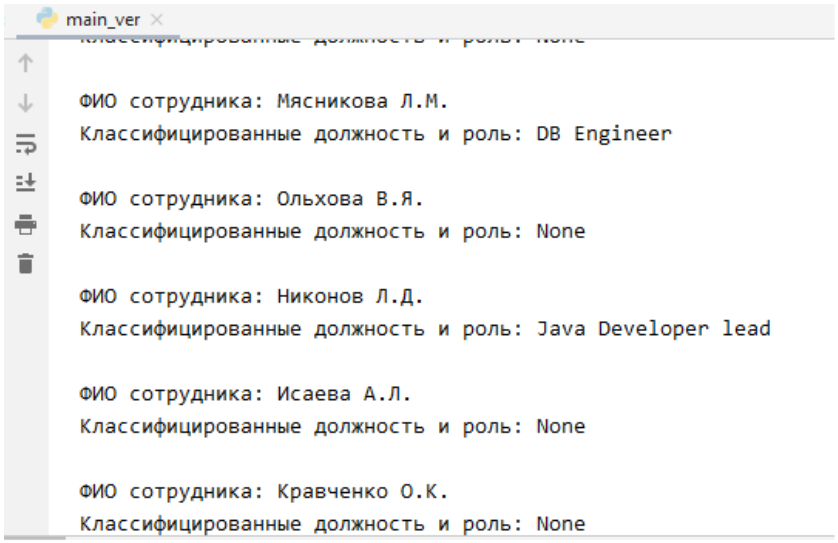


Рис. 4. Результат работы ЭС

Заключение

Итогом данной работы является реализованная экспертная система, которая при правильном заполнении базы знаний способна оказать большую поддержку при принятии решений о найме новых сотрудников или в моменты кадровых перестановок.

Литература

1. *Сергиенко, М. А.* О некоторых свойствах базы нечетких продукционных правил / М. А. Сергиенко // Международный научно-исследовательский журнал. – 2015. – Ч. 3, № 1(32). – С. 27–28.

2. *Джарратано, Д.* Экспертные системы: принципы разработки и программирование / Д. Джарратано, Г.Райли. – 4-е изд. – М. : Издательский дом «Вильямс», 2007. – 1152 с.

3. Guide to the Project Management Body of Knowledge PMBOK Guide 6th Edition, Newtown Square, Pa: Project Management Institute, 2018 – 41c.

4. SciKit-Fuzzy – skfuzzy v0.2 docs [Электронный ресурс]. – URL: https://pythonhosted.org/scikit-fuzzy/overview.html

Данилова Ирина Владимировна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: irina0danilova@gmail.com

Сергиенко Михаил Александрович (научный руководитель) – канд. техн. наук, доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: sergienko-m-a@yandex.ru

УДК 519.688

ОПРЕДЕЛЕНИЕ КООРДИНАТ ОБЪЕКТА В ЛОКАЛЬНОМ ПОМЕЩЕНИИ ПО ВИДЕОПОТОКУ

В. И. Девятов

Воронежский государственный университет

Введение

Существует множество устройств, которым необходимо определять положение в пространстве для выполнения той или иной работы. За этот процесс могут отвечать разнообразные датчики. Однако иногда их точности может быть недостаточно, и прибор может некорректно определять некоторые окружающие его предметы. Например, робот-пылесос может не замечать тонкие ножки стульев, что может вызывать неудобства. Для усовершенствования работы таких умных устройств может применяться компьютерное зрение.

В данной статье предлагаются подходы к нахождению точного положения объектов в помещении в режиме реального времени с помощью видеокамер.

1. Постановка задачи

Пусть имеется некоторое помещение с определенным образом установленными видеокамерами (рис. 1). Для него известны точные размеры. Поставленная задача состоит в том, чтобы определить координаты некоторого объекта, находящегося в этом помещении, при помощи видеопотока с камер.

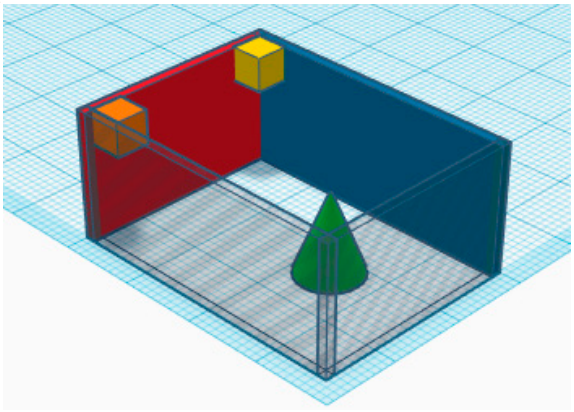


Рис. 1. Помещение с камерами и объектом

2. Методы решения

Для решения подобных задач в первую очередь должны применяться алгоритмы распознавания объектов в видеопотоке [3]. Методы распознавания образов состоят в разбиении рассматриваемой совокупности объектов на некоторое количество классов по следующему принципу: объекты, отнесённые к одному классу, должны быть схожи по некоторому критерию, а объекты из разных классов должны иметь существенные различия по тому же или другому критерию. После получения такого разбиения становится возможным распознать объект [1].

Поставленную задачу нахождения координат объекта предлагается решать двумя способами.

Первый способ – расположить одну камеру в одном из углов комнаты (рис. 2).

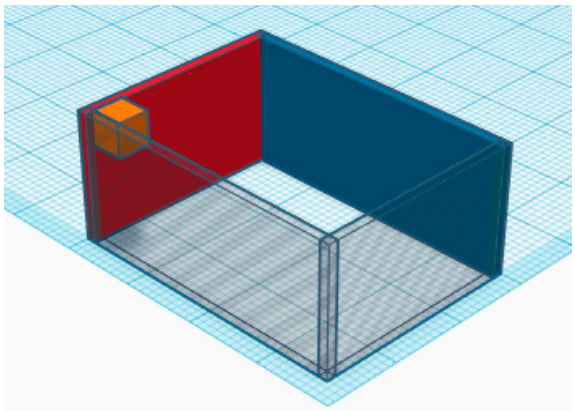


Рис. 2. Вариант расположения камеры в первом способе

Используя эту камеру, с помощью алгоритмов распознавания образов необходимо обнаружить искомый предмет. Также необходимо выделить контуры двух смежных стен. Далее, выпуская фиктивные лучи параллельно одной из стен, найти первый и последний лучи, которые пересекают данный объект в кадре. Основываясь на специальных метках на стене, возможно определить координаты объекта. Далее необходимо повторить данный процесс для второй стены. Таким образом, будут получены координаты объекта на полу комнаты.

Недостатками данного способа являются распознавание не только объекта, но и контуров стен, а также необходимость трассировок большого количества лучей.

Второй способ – взять две камеры, установить их рядом друг с другом и расположить сверху посередине стены (рис. 3). Данный способ основывается на идеях бинокулярного зрения [2]. Бинокулярным зрением называется способность зрения формировать единый образ из изображений, полученных с двух глаз (в данном случае, с двух камер).

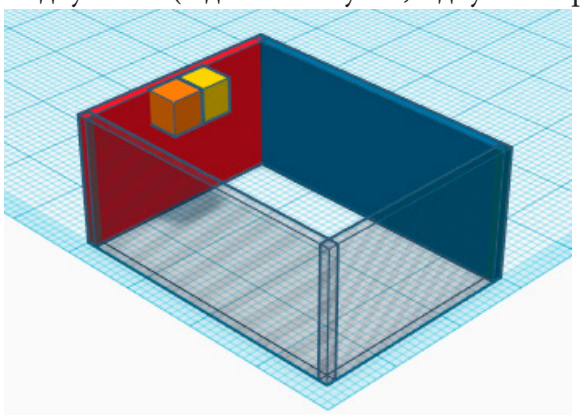


Рис. 3. Вариант расположения камер при втором способе

Далее, аналогично первому способу, необходимо распознать искомый объект на двух видеопотоках. После выпуска лучей получим ситуацию, которую иллюстрирует рис. 4.

Из подобия треугольников можно получить следующие формулы:

$$\frac{P_L}{f} = \frac{-\frac{1}{2}b + x}{Z}, \tag{1}$$

$$\frac{P_R}{f} = \frac{\frac{1}{2}b + x}{Z}. \tag{2}$$

Применив одну из описанных выше формул, возможно вычислить координаты предмета.

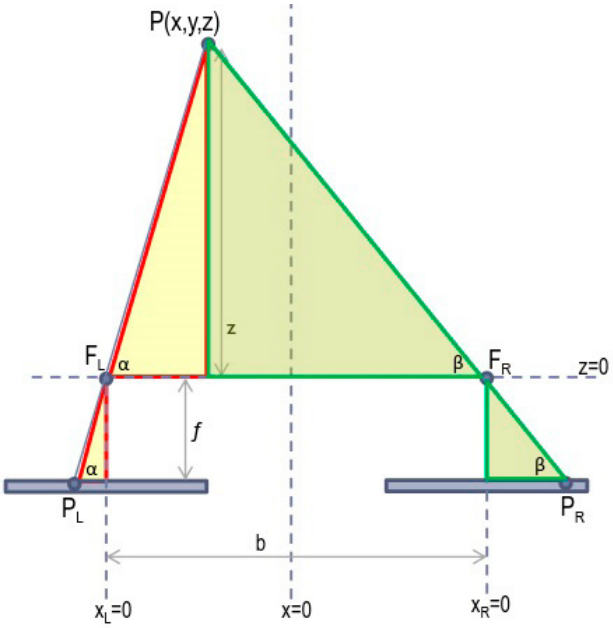


Рис. 4. Выпуск лучей на предмет из двух камер

Достоинство данного подхода в том, что расширяется поле зрения. Недостатки данного способа проявляются в нахождении небольшого количества точек объекта, с помощью которых можно измерить расстояние, а так же в более сложном поиске.

Заключение

На текущий момент оба подхода к решению задачи рассмотрены теоретически. Изучены и реализованы алгоритмы распознавания объектов по видеопотоку. В дальнейшем предполагается попытаться реализовать оба способа на практике и подробнее рассмотреть их преимущества и недостатки и выбрать наиболее подходящий вариант.

Литература

1. *Мазуров, Вл. Д.* Математические методы распознавания образов : учебное пособие / Вл. Д. Мазуров. – 2-е изд., перераб. и доп. – Екатеринбург : Изд-во Урал. ун-та, 2010. – 101 с.

2. *Шеломенцева, И. Г.* Оценка восприятия глубины разными видами машинного зрения [Электронный ресурс] / И. Г. Шеломенцева, Н. В. Якасова // Современная техника и технологии: науч.-прак. журн. – 2015. – № 11. – Режим доступа: http://technology.snauka.ru/2015/11/8287 (Дата обращения: 22.04.2020).

3. *Зенин, А. В.* Анализ методов распознавания образов [Электронный ресурс] / А. В. Зенин // Молодой ученый. – 2017. – № 16 (150). – С. 125–130. – Режим доступа: https://moluch.ru/archive/150/42393/ – (Дата обращения: 23.04.2020).

Девятов Владислав Игоревич – магистрант 1-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: d3vyatov@gmail.com

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доцент, доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 618

АНАЛИЗ МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ МАШИННОГО ОБУЧЕНИЯ

А. М. Джалолов

Воронежский государственный университет

Введение

Целью работы является создание модели для выявления метастатического рака на небольших участках изображения, взятых из цифровых сканирований патологии.

1. Общие сведения

1.1. Классическое машинное обучение

Для понимания дальнейшего повествования необходимо понимание отличия терминов.

Машинное обучение (англ. machine learning, ML) – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение в процессе применения решений множества сходных задач. Для построения таких методов используются средства математической статистики, численных методов, методов оптимизации, теории вероятностей, теории графов, различные техники работы с данными в цифровой форме.

Различают несколько областей машинного обучения, в зависимости от литературы:

- Обучение с подкреплением (англ. Reinforcement Learning)
- Обучение с учителем (англ. Supervised Learning).
- Обучение с частичным привлечением учителя(англ. Semi-supervised Learning)
- Обучение без учителя (англ. Unsupervised Learning).

В некоторой литературе обучение с подкреплением относится к обучению без учителя, но сейчас все чаще их отличают друг от друга. Здесь будет рассмотрено только обучение с учителем, которое будет использовано в данном исследовании, однако в разделе «Литература» есть ссылки и на другие классы алгоритмов, чтобы желающие могли ознакомиться.

Обучение с учителем является самой популярной областью машинного обучения, зачастую при помощи нее решаются два класса задач: классификация и регрессия. Как правило имеется некое наблюдение за объектом и результат этого наблюдения, который выражается либо в классификации поведения, либо в числовом значении соответственно. Главное преимущество данной области по отношению к двум другим, является получение наиболее точных прогнозов, однако здесь есть и свой изъян. Данные, на которых обучается модель должны быть размечены человеком, и в последствии нейронная сеть имитирует поведение человека.

Машинное обучение является довольно широким понятием, в которое входит **глубокое обучение**, которое будет рассмотрено ниже.

1.2. Глубокое обучение

Глубокое обучение (глубинное обучение; англ. Deep learning) – совокупность методов машинного обучения (с учителем, с частичным привлечением учителя, без учителя, с подкреплением), основанных на обучении представлениям (англ. feature/representation learning), а не специализированным алгоритмам под конкретные задачи.

Основное отличие данного семейства методов от классического машинного обучения в нейронных сетях, с появлением которых возрастает возможность прогнозирования сложного поведения объекта.

2. Анализ набора данных

Для исследования был взят датасет PatchCamelyon (PCam), который составил Karl Pearson. Данный набор данных представляет собой 327680 изображений размером 96x96 пикселей, полученных из сканирований срезов лимфатических узлов. В каждом изображении имеется двоичная метка, указывающая на наличие метастатической ткани. Основной задачей является автоматизировать процесс предсказания наличия метастатической ткани на снимке.

После удаления дубликатов распределение изображений по меткам стало 60/40 (отрицательные/положительные) как показано на рис. 1.

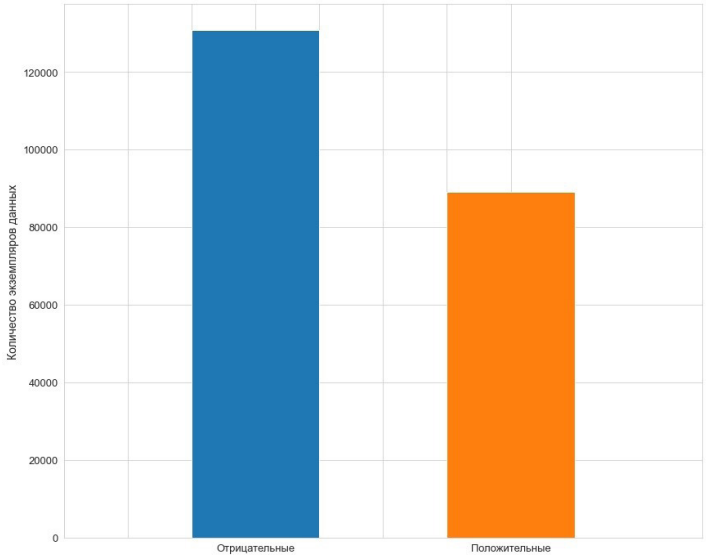


Рис. 1. Распределение примеров по классам

Отрицательная матка означает, что в центральной области (32×32 пикселя) изображения имеется по крайней мере один пиксель опухолевой ткани. Однако нельзя обрезать их до размера центральной области, поскольку в граничных областях хранится много информации, по мнению автора данных.

3. Метрики

Для получения качественной модели требуется определить функцию качества и функцию потерь.

В данной задаче классификации за функцию потерь будет использована бинарная кросс-энтропия:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p(y_i)) + (1 - y_i)\log(1 - p(y_i)).$$

Необходимо понимать как определить, что модель хорошо обучилась. Первое, что приходит на ум это процент правильно предсказанных изображений. Данная метрика носит незамысловатое имя – точность. Однако у этого подхода есть изъян, а именно несбалансированные выборки. В нашем случае, если будет построена модель предсказывающая всегда отрицательную метрику, то точность будет 60 процентов, хотя такая модель очевидно никого не устроит. Чтобы не попасть в эту ловушку была придумана метрика roc_auc.

Для понимания метрики roc_auc, необходимо ввести некоторые новые термины:

TP – количество примеров, где модель предсказала положительную метку, и фактическая метка была также положительной.

TN – количество примеров, где модель предсказала отрицательную метку, и фактическая метка была также отрицательной.

FP – количество примеров, где модель предсказала положительную метку, но фактическая метка была отрицательной.

FN – количество примеров, где модель предсказала отрицательную метку, но фактическая метка была положительной.

TPR – доля примеров, которых модель правильно отнесла к положительному классу, по отношению ко всем фактическим позитивным меткам в данных. (1)

FPR – доля примеров, которых модель неверно отнесла к положительным, по отношению ко всем фактическим отрицательным меткам. (2)

$$TPR = \frac{TP}{FN + TP} \quad (1), \quad FPR = \frac{FP}{FP + TN} \quad (2)$$

И наконец, roc_auc это площадь под кривой (ROC curve), построенной относительно координат TPR и FPR, которые считаются относительно разных границ (англ. threshold) вероятности модели. Модель на выходе дает не просто метку класса, а вероятность принадлежности к классу, что означает уверенность нейронной сети в своем решении. Эту вероятность и меняют для получения roc_auc. Таким образом метрика рассмотренная выше позволяет показать не только точность, но и то насколько уверена модель. На рис. 2 показан пример использования roc_auc для анализа точности нейронной сети. Зачастую помимо построения ROC_curve для модели, также на графике показывают функцию $y = x$, отражающую значение данной метрики в случае, когда модель предсказывает класс случайно, в этом случае roc_auc будет равен 0.5. Очевидно, что если модель показывает не сильно лучший результат, то она бесполезна.

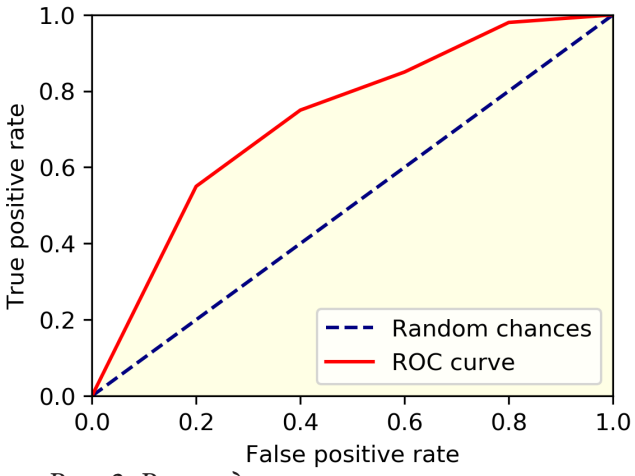


Рис. 2. Распределение примеров по классам

4. Модели

4.1. Модель с нуля

Для начала была построена модель с нуля, что делают не часто, так как это требует огромных ресурсов и большого количества данных. В архитектуре нейронной сети использовался стандартный подход к задачам классификации изображений – чередование сверточного слоя, нормализации, выделения признаков и регуляризации (прилож. 1).

Результатом обучения стала точность 90 процентов, ROC curve показана на рис. 3

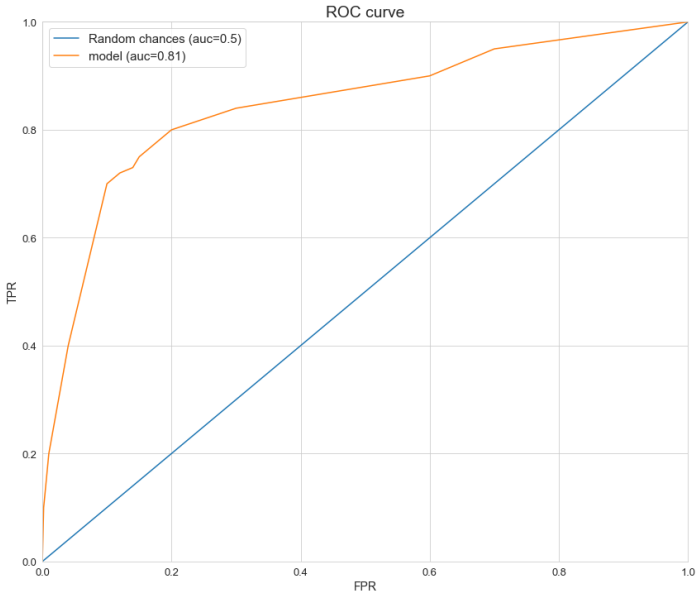


Рис. 3. ROC curve

Как видно из графика и метрики точности, можно сделать вывод, что модель дает предсказания лучше, чем случайный выбор. Однако стоит попробовать улучшить точность модели, использовав более современные подходы машинного обучения.

4.1. Использование предобученной модели (англ. Transfer Learning)

В современных задачах все чаще используют предобученные модели, так как первые сверточные слои выделяют графические примитивы, например, изменение цветов, что сигнализирует, о границах объектов на изображении, затем идут линии, и так далее. Таким образом, часто используют сверточные слои уже готовых нейронных сетей, потому что разницы, где обучать модель распознавать линии или цветовую гамму нет.

Было решено использовать модель densenet169 (прилож. 2), которая обернута в фреймворк fastai. Использование fastai дает более быструю сходимость модели за счет оптимизированного использования графического процессора. Также в этом фреймворке находится много полезных методов, позволяющих писать меньше кода, как следствие меньше ошибок.

В данной модели обучался только последний слой, остальные были заморожены. Причиной этому служит недостаток вычислительной мощности. Результатом такого подхода к прогнозированию задачи стала точность 95 процентов и график ROC curve, показанный на рис. 4.

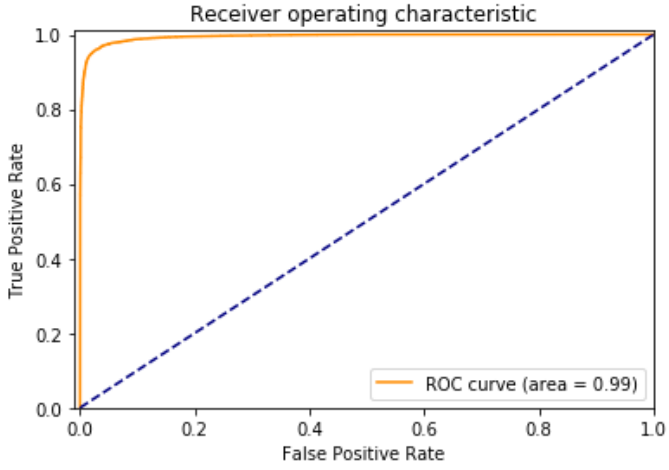


Рис. 4. ROC curve fastai

Результат впечатляет, однако можно было бы получить большую точность, если обучать больше слоев в модели. Соответственно задачу можно масштабировать дальше, имея необходимые ресурсы.

Заключение

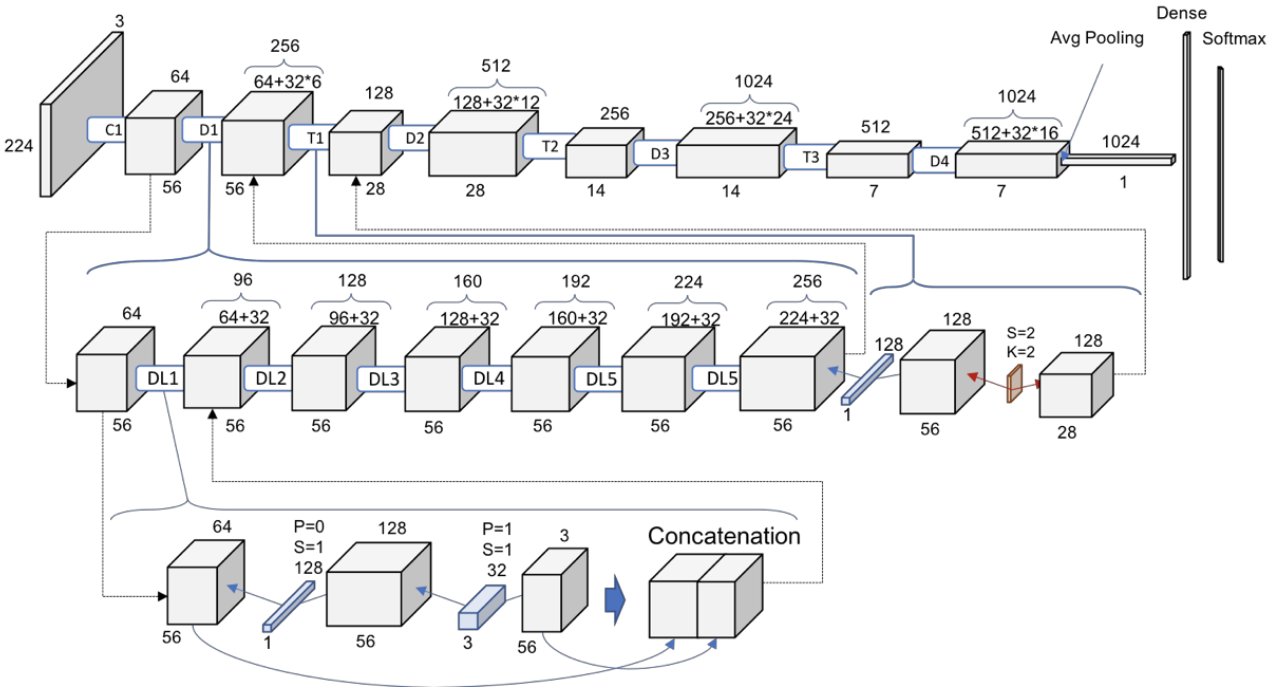
В ходе исследования получена модель способная предсказывать с точностью 95 процентов метастатический рак на изображении. Благодаря этому можно автоматизировать рутинные задачи врачей и убрать человеческий фактор при постановке диагноза. Однако стоит упомянуть, точность модели напрямую зависит от качества данных.

Приложения

Layer (type)	Output Shape	Param #
conv2d_19 (Conv2D)	(None, 94, 94, 32)	896
conv2d_20 (Conv2D)	(None, 92, 92, 32)	9216
batch_normalization_19 (Batc	(None, 92, 92, 32)	128
activation_19 (Activation)	(None, 92, 92, 32)	0
max_pooling2d_10 (MaxPooling	(None, 46, 46, 32)	0
dropout_13 (Dropout)	(None, 46, 46, 32)	0
conv2d_21 (Conv2D)	(None, 44, 44, 64)	18432
batch_normalization_20 (Batc	(None, 44, 44, 64)	256
activation_20 (Activation)	(None, 44, 44, 64)	0
conv2d_22 (Conv2D)	(None, 42, 42, 64)	36864
batch_normalization_21 (Batc	(None, 42, 42, 64)	256
activation_21 (Activation)	(None, 42, 42, 64)	0
max_pooling2d_11 (MaxPooling	(None, 21, 21, 64)	0
dropout_14 (Dropout)	(None, 21, 21, 64)	0
conv2d_23 (Conv2D)	(None, 19, 19, 128)	73728
batch_normalization_22 (Batc	(None, 19, 19, 128)	512
activation_22 (Activation)	(None, 19, 19, 128)	0
conv2d_24 (Conv2D)	(None, 17, 17, 128)	147456
batch_normalization_23 (Batc	(None, 17, 17, 128)	512
activation_23 (Activation)	(None, 17, 17, 128)	0
max_pooling2d_12 (MaxPooling	(None, 8, 8, 128)	0
dropout_15 (Dropout)	(None, 8, 8, 128)	0
flatten_4 (Flatten)	(None, 8192)	0
dense_7 (Dense)	(None, 256)	2097152
batch_normalization_24 (Batc	(None, 256)	1024
activation_24 (Activation)	(None, 256)	0
dropout_16 (Dropout)	(None, 256)	0
dense_8 (Dense)	(None, 1)	257

Total params: 2,386,689
Trainable params: 2,385,345
Non-trainable params: 1,344

Приложение 1. Архитектура нейронной сети для модели из раздела 4.1



Приложение 2. Архитектура нейронной сети для модели из раздела 4.2

Литература

1. *Николенко С., Кадурин А., Архангельская Е.* Глубокое обучение. – СПб. : Питер, 2018. – 480 с. (Серия «Библиотека программиста»).

2. Документация по фреймворку fastai. – Режим доступа: https://docs.fast.ai

3. *Cortes C., Gonzalvo X., Kuznetsov V., Mohri M. and Yang S.* Adanet: Adaptive structural learning of artificial neural networks. arXiv preprint arXiv:1607.01097, 2016.

4. *Huang G., Liu Z., Maaten L.* Densely Connected Convolutional Networks. arXiv:1608.06993v5, 2018

5. *Gatys L., Ecker A. and Bethge M.* A neural algorithm of artistic style. Nature Communications, 2015.

6. *Romero A., Ballas N., Kahou S. E., Chassang A., Gatta C. and Bengio Y.* Fitnets: Hints for thin deep nets. In ICLR, 2015.

Джалолов Алишер Махмадсаидович – студент 3-го курса кафедры Математического обеспечения ЭВМ Воронежского государственного университета. E-mail: dzhalolov.alik@gmail.com

Светлана Юрьевна Болотова (научный руководитель) – канд. физ.-мат. наук, доцент кафедры Математического обеспечения ЭВМ Воронежского государственного университета. E-mail: bolotova.svetlana@gmail.com

УДК 336.71

МЕТОДЫ УПРАВЛЕНИЯ ЛИКВИДНОСТЬЮ КОММЕРЧЕСКОГО БАНКА

М. В. Жабина, Н. Б. Баева

Воронежский государственный университет

Введение

В наши дни в России наблюдается неустойчивая экономическая ситуация. Это происходит по ряду причин: политика государства, снижение уровня национального производства и быстрый рост импорта товаров, влияние цен на нефть и газ, развитие коррупции и монополии. Вследствие чего, происходят закрытия предприятий, увольнения, падение спроса на рынке продаж. Также множество последствий обрушилось и на финансовый сектор. Например, при инфляции денег кредиторы терпят убытки, потому что им возвращают гораздо меньше, чем заняли. Нестабильный курс и падение рубля сильно отразилось на функционировании банковского сектора. Исходя из всего вышесказанного, становится понятно, что на сегодняшний день актуален вопрос оптимизации кредитных портфелей. Исходя из этого, были проанализированы варианты достижения максимальной прибыли кредитного портфеля с помощью учета норм ликвидности, которые установлены Центральным Банком России, а также непосредственное удовлетворение портфеля данным нормам.

Под понятием ликвидности банка подразумевается способность банка обеспечить своевременное и полное исполнение своих обязательств.

Прежде чем описывать процессы, управлением ликвидностью коммерческого банка, необходимо внести ясность и разграничить такие понятия как: ликвидность активов, ликвидность баланса кредитной организации, ликвидность коммерческого банка.

1. Ликвидность активов банка

Ликвидность активов банка – это способность активов быстро и без существенных потерь трансформироваться в деньги посредством их реализации или исполнения обязательств должниками; степень возможных потерь также обусловливается качеством активов.

Баланс банка считается ликвидным, если его состояние позволяет за счет быстрой реализации средств по активу покрывать срочные обязательства по пассиву. Ликвидность коммерческого банка это, прежде всего, динамическое состояние, т. е. ликвидность характеризуются не статическими показателями баланса, а способностью организации обеспечивать равновесие проходящих через него денежных потоков без ущерба для рентабельности банка.

Ликвидность банка тесно связана с ликвидностью баланса. В целях поддержания ликвидности баланса банк обязан постоянно поддерживать необходимый и достаточный уровень средств на корреспондентских счетах, наличных средств в кассе, быстрореализуемых активов, т.е. управлять ликвидностью.

Основными элементами по управлению ликвидностью являются: анализ состояния мгновенной, текущей и долгосрочной ликвидности, составление краткосрочного прогноза ликвидности, проведение анализа ликвидности и использование негативного для банка развития событий (состояние рынка, положение заемщиков и кредиторов), определение потребности банка в ликвидных средствах, определение избытка/дефицита ликвидности и предельно до-

пустимых его значений, оценка влияния на состояние ликвидности операций в иностранной валюте, определение предельных значений коэффициентов ликвидности по каждой валюте и по всем валютам в целом.

Оценка ликвидности банка является одной из наиболее сложных задач, позволяющих получить ответ на самый важный вопрос: способен ли банк отвечать по своим обязательствам. На способность банка отвечать по обязательствам влияют характеристики состояния и изменения ресурсной базы, возвратность активов, финансовый результат деятельности, размер собственных средств (капитала) банка, а также качество управления банком, менеджмент, которые в определенные моменты могут сыграть и играют решающую роль.

Для контроля за состоянием ликвидности банка установлены три норматива ликвидности (мгновенной, текущей и долгосрочной). Они определяются как соотношение между активами и пассивами с учетом сроков, сумм и видов активов, а также других факторов.

2. Основные нормативы ликвидности

Норматив мгновенной ликвидности регулирует (ограничивает) риск потери банком ликвидности в течение одного операционного дня.

Рассчитывается по формуле:

$$H_2 = \frac{\sum_{i=1}^{k} L_1 \bullet x_{1i}}{\sum_{j=1}^{k} O_1 \bullet y_{1j}} \geq \frac{15}{100} \qquad (1)$$

и определяет минимальное отношение суммы высоколиквидных активов банка к сумме пассивов банка по счетам до востребования.

Норматив текущей ликвидности регулирует (ограничивает) риск потери банком ликвидности в течение ближайших к дате расчета норматива 30 календарных дней и определяет минимальное отношение суммы ликвидных активов банка к сумме обязательств (пассивов) банка по счетам до востребования со сроком исполнения обязательств в ближайшие 30 календарных дней, скорректированных на величину минимального совокупного остатка средств по счетам физических и юридических лиц (кроме кредитных организаций) до востребования и со сроком исполнения обязательств в ближайшие 30 календарных дней. Рассчитывается по формуле:

$$H_3 = \frac{\sum_{i=1}^{k} L_2 \bullet x_{2i}}{\sum_{j=1}^{k} O_1 \bullet y_{2j}} \geq \frac{50}{100}. \qquad (2)$$

К высоколиквидным и ликвидным активам относятся только те финансовые активы банка, которые в соответствии с нормативными документами Банка России относятся к первой категории качества (1-й группе риска) и второй категории качества (2-й группе риска). Кроме вышеперечисленных активов в расчет показателей включаются остатки на балансовых счетах, по которым отсутствуют требования по формированию резервов, в случае, если активы, числящиеся на соответствующих балансовых счетах, планируются банком к получению в течении тридцати ближайших календарных дней в форме, позволяющей отнести их к высоколиквидным и ликвидным активам.

Норматив долгосрочной ликвидности регулирует (ограничивает) риск потери банком ликвидности в результате размещения средств в долгосрочные активы и определяет максимально

допустимое отношение кредитных требований банка с оставшимся сроком до даты погашения свыше 365 или 366 календарных дней, к собственным средствам (капиталу) банка и обязательствам (пассивам) с оставшимся сроком до даты погашения свыше 365 или 366 календарных дней, скорректированным на величину минимального совокупного остатка средств по счетам со сроком исполнения обязательств до 365 календарных дней и счетам до востребования физических и юридических лиц (кроме кредитных организаций). Рассчитывается по формуле:

$$\acute{I}_4 = \frac{\sum_{i=1}^{k} L_3 \bullet x_{3i}}{C + \sum_{j=1}^{k} O_3 + y_{3j}} \le \frac{120}{100}. \quad (3)$$

Минимально допустимое значение норматива долгосрочной ликвидности устанавливается Банком России в размере 120 %.

Нарушение норм мгновенной ликвидности и текущей ликвидности говорит о недостаточном запасе ликвидности у кредитной организации. Несоблюдение норм долгосрочной ликвидности говорит о том, что банк злоупотребляет размещением в долгосрочные активы краткосрочных пассивов. Неисполнение нормативов может быть чревато штрафными санкциями со стороны Банка России, введением запрета на осуществление определенных банковских операций, а в случае многократных нарушений вообще привести к отзыву лицензии. Впрочем, в отдельных случаях Центральный Банк может изменить на срок до шести месяцев установленные значения нормативов для банка-нарушителя.

Заключение

Наличие эффективной банковской системы – важная черта любой современной рыночной экономики. Основными целями развития нынешнего банковского сектора является: укрепление его устойчивости, исключающее возможность возникновения системных банковских кризисов; повышение качества осуществления банками функций по аккумулированию денежных средств населения, предприятий и их трансформации в кредиты и инвестиции; укрепление доверия к российскому банковскому сектору со стороны инвесторов, кредиторов и вкладчиков и самое главное населения. Как показывает практика мировая финансовая и банковская системы подвержены периодическим кризисам. В этой связи, с учетом анализа возникающих кризисных ситуаций, необходимо уметь быстро подстраиваться под изменяющийся рынок и экономику и иметь дифференцированные портфели финансовых операций, потому что чем выше дифференциация, тем меньше риск стать банкротом.

Одной из важных причин неустойчивости работы банков является проведение финансовых операций с риском и поэтому в данной работе рассмотрен портфель с высокой степенью риска активов и низкой степенью ликвидности. Была максимизирована прибыль портфеля и минимизированы риски. Также был выявлен путь получения наибольшей выгоды от заемщиков при минимальных рисках, а портфель был проверен на нормы ликвидности ЦБР. Для реализации моделей разработаны алгоритмы и программы, а также проведены экспериментальные расчеты.

С использованием нормативов ликвидности строилась модель, алгоритм и программа отлаживались на экспериментальных данных и в условиях одного предприятия в г. Воронеж. Вместе с тем полученная модель, программа и алгоритм достаточно универсальны и могут быть использованы для повышения эффективности работы любого коммерческого банка.

Литература

1. *Азарнова, Т. В.* Математические модели производственных процессов, логистики и риска: учебное пособие для вузов / Т. В. Азарнова, Н. Б. Баева. Воронеж: Издательско-полиграфический центр Воронежского гос. ун-та, 2013. – 137 с.

2. *Белоглазова, Г. Н.* Деньги. Кредит. Банки: Учебник / Под редакцией Г. Н. Белоглазовой, Белоглазова Г. Н. – М.: Высшее образование, 2009. – 392 с.

3. *Баева, Н. Б.* Основы теории систем и вычислительные схемы системного анализа: методическое пособие для вузов / Н. Б. Баева, Д. В. Ворогушина, Е. В. Куркин. Воронеж: Издательско-полиграфический центр Воронежского гос. ун-та, 2011. – 76 с.

Жабина Мария Владимировна – магистрант 2-го года обучения кафедры математических методов исследования операций Воронежского государственного университета. E-mail: marizhabo@gmail.com

Баева Нина Борисовна (научный руководитель) – д-р физ.-мат. наук, проф., профессор кафедры математических методов исследования операций Воронежского государственного университета. E-mail: annabaeva73@gmail.com

УДК 004.032.26

РЕАЛИЗАЦИЯ МЕТОДОВ КОЛИЧЕСТВЕННОЙ МЕТАЛЛОГРАФИИ С ПОМОЩЬЮ ГЛУБОКИХ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ

М. Н. Жихарева

Воронежский государственный университет

Введение

Простейшим видом количественного анализа является визуальная оценка структуры: «мельче, крупнее, однородное или нет и насколько». В современной металлографии этого не достаточно. Необходимы точные количественные оценки для того, чтобы проследить кинетику изменения структуры в процессе внешнего воздействия (термического, механического и т. д.) и определить механизмы реализующихся процессов.

Это подталкивает к поиску новых решений для анализа структур. В то же время на протяжении последних лет активно развиваются нейросетевые технологии для решения ряда задач компьютерного зрения, поэтому применение нейросетей в металлографии вполне оправдана.

Задача поставлена компанией НЛМК. На текущий момент ее решением занимается оператор, и есть стремление автоматизировать процесс.

1. Постановка задачи

Разработать систему машинного зрения, способную:
– сегментировать исходное изображение;
– автоматически размечать изображение по нескольким критериям:
– площадь зерен;
  – периметр
  – эквивалентный диаметр
  – фактор формы
– строить гистограммы распределений перечисленных величин;
– раскрашивать исходное изображение согласно гистограмме.

Критериями качества разрабатываемого решения служит расстояние Вассерштейна [1] от истинного распределения соответствующей метрики до полученной.

2. Анализ задачи

ГОСТы по определению балла зерна, количества перлита, величины включений (например графита в чугуне) и т.п. являются количественной металлографией. Стандартный подход предполагает переход от общего к частному – создание эталонов структур (которые имеют количественную оценку), которыми можно пользоваться для количественной оценки структур изучаемых материалов. Это усредненные параметры, величина которых, как правило, связана с понятием «балл» конкретной структуры. Иных вариантов количественного анализа на момент разработки ГОСТов не было. В лабораторной практике проводились количественные оценки структуры путем примитивных линейных измерений, планиметрирования и т. п. Все эти методы давали адекватные результаты, но были исключительно трудоемкими [2].

Современные программы обработки изображений позволяют определить площадь и линейные размеры зерна любой фазы, периметр, а также производные от этих величин – фактор формы, средний размер, эквивалентный диаметр, и т. д. Развитие компьютерных методов дает возможность измерить непосредственно геометрические параметры каждой структурной единицы, обработать результаты и получить не только среднее, но и ряд определенных зависимостей – распределений определенной величины (размера зерна, диаметра и пр.) по размерам (частотная кривая).

2.1. Металлографические метрики

Определение *площади* объектов в программе обработки изображений – это наиболее объективный вид анализа. Анализируемый объект «состоит» из пикселов изображения. А поскольку метрическая часть программы откалибрована, то каждый пиксел имеет площадь. Количественно эта площадь определяется увеличением при съемке и разрешением камеры (или фотоаппарата). Таким образом, определение площади в программе обработки изображений – это суммирование площадей всех пикселов, составляющих объект (принадлежащих объекту)

Для зерна металла или сплава *эквивалентный диаметр* – это диаметр круга, площадь которого равна площади зерна. То есть зерно в данном случае считается круглым. Понятие «диаметр зерна» используется в ГОСТ 5639-82 [3].

Минимальный и максимальный диаметр – это длины сторон минимального окаймляющего объект прямоугольника, стороны которого параллельны границам изображения.

Длина и ширина – это длины сторон окаймляющего объект прямоугольника, одна из которых параллельна оси симметрии объекта. Приблизительное равенство распределений по этим размерам означает, что зерна имеют равноосную форму.

Периметр – это количество граничных пикселей объекта. При наличии калибровки пиксел имеет размер и периметр зерна можно получить в мкм или в см. Периметр – это тот параметр структуры, который невозможно (или крайне затруднительно) определить вручную, и тем ценнее применение программ обработки изображений. Если известен периметр, можно определить фактор формы.

Фактор формы определяется по формуле:

$$F = \frac{4\pi S}{P^2},$$

где S – площадь объекта, а P – периметр. Чем меньше многоугольник похож на круг, тем меньше отношение его площади к периметру. У квадрата, к примеру, фактор формы равен 0.785, а у равностороннего треугольника – 0.605.

2. Реализация

2.1. Разметка исходных данных

Продольный и поперечные шлейфы были размечены вручную (рис. 1) и разделены на выборки: продольный был взят как источник данных для обучающей выборки, а поперечный – для валидационной, соответственно. При этом в эталонной разметке не были размечены те границы между ячеек, которые не видны: очевидно, что если они не видны человеку, то и нейросеть не сможет их воссоздать.

С другой стороны, все ячейки по форме должны образовывать выпуклые многоугольники, и, если этого не наблюдается, то ячейки должны быть принудительно разделены. Для автоматизированного решения этой задачи был использован алгоритм водораздела, который будет описан далее.

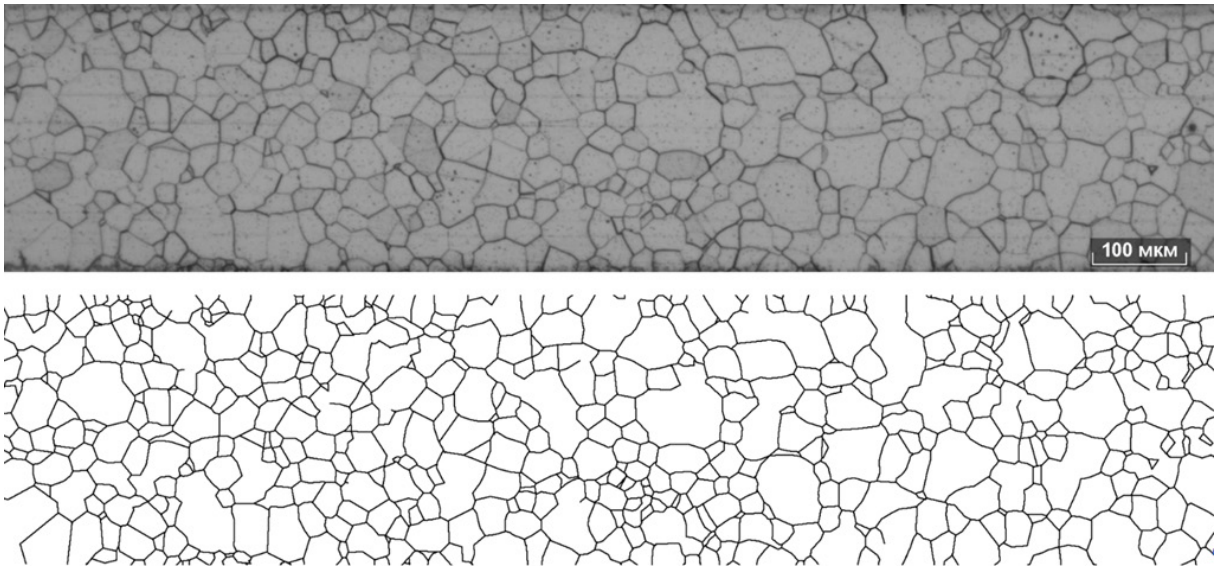


Рис. 1. Размеченное изображение

2.2. Оценка U-Net-решения

В рамках решения задачи сегментации очень хорошо себя зарекомендовала сеть U-Net [4], которая и сейчас пользуется популярностью при решении различных задач.

Причина популярности сети в том, что она полностью сверточная, в ней нет полносвязных слоев: только Convolution, UpConvolution, MaxPooling и BatchNormalization. Она проста концептуально, плохо переобучается и из-за относительного небольшого количества параметров ей нужно меньше данных для обучения. Структура сети показана на рис. 2.

Использованные размеры слоев: 16–32–64–128–256–128–64–32–16 (в оригинальной статье

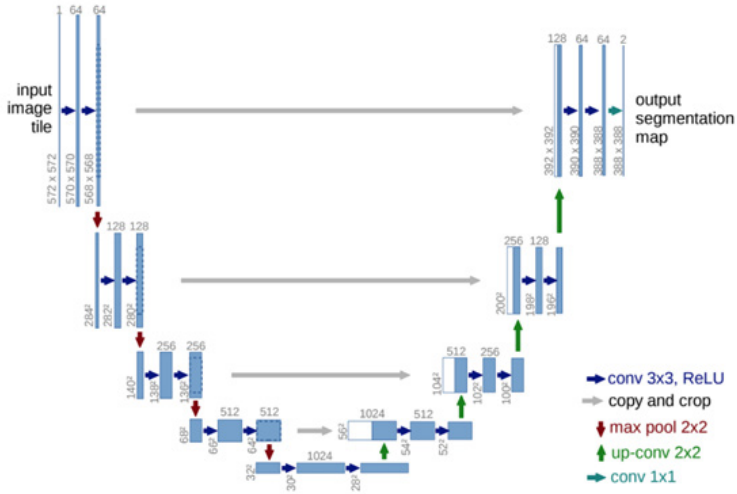


Рис. 2. Структура сети U-Net

вместо 16 – 64, и соответственно размеры остальных слоев удваиваются). Итого сеть имеет 839489 обучаемых параметра.

Данные для сети подготовлены следующим образом:

1. Из исходного изображения, взятого как источник данных для обучающей выборки, вырезаются случайные сегменты размером 200×200.

2. К ним применяется аугментации случайного поворота на 90 и отражений, горизонтальных и вертикальных.

3. Изображения увеличиваются в 2 раза и становятся уже 400×400. Это сделано для того, чтобы линии маски, изначально нарисованные при разметке толщиной 1 пиксель, стали толще, и рецептивным полям сверточных слоев было проще видеть присутствие границы-разделителя между ячейками металла. Как альтернативный подход можно было, не увеличивая разрешения, применить морфологическую операцию эрозии к сегменту маски, а, взяв предсказание сети, выполнить обратную операцию – дилатацию.

4. Итоговый батч нормализуется делением значений интенсивности на 255 (итоговый размах – от 0 до 1, как на входе сети, так и на выходе).

Другие параметры эксперимента:

– Размер батча: 16

– Количество эпох: 30

– Количество батчей на 1 эпоху при обучении: 50

– Количество батчей на 1 эпоху при валидации: 10

– Оптимизатор: Adam

– Функция потерь: взвешенная MSE

Классическая MSE представляет собой следующую функцию [5]:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2.$$

Под «взвешенным MSE» подразумевается следующая функция потерь:

$$WMSE = \frac{1}{N}\sum_{i=1}^{N}(\alpha[y_i - \hat{y}_i < 0](y_i - \hat{y}_i)^2 + \beta[y_i - \hat{y}_i \geq 0](y_i - \hat{y}_i)^2),$$

здесь y_i – правильный ответ, $\hat{y}_i$ – предсказанная величина, [ ] – скобка Айверсона, α – коэффициент ошибки первого рода, β – коэффициент ошибки второго рода.

В эксперименте эмпирически принято $\alpha = 1.5$. Это значит, что каждый раз, когда нейросеть предсказывала 1 (белый пиксель) там, где должен быть 0 (черный пиксель), данная разность вносилась в функционал с «весом» не 1, а $1.5^2 = 2.25$. С одной стороны, это дало больше ложно-отрицательных предсказаний (черные точки в ячейках), с другой – линии на границах стали более ярко выраженными.

2.3. Получение центров ячеек

Сначала получены точки, соответствующие центрам ячеек. Сделано это с помощью следующего алгоритма:

Итеративно в цикле:

1. Применяется операция эрозии;

2. Находятся контуры на изображении-результате;

3. Если контуров не нашлось (из-за того, что вся картинка залита черным), то выходим из цикла;

4. Если площадь меньше пороговой (равной 200, определено эмпирически), то заносим в массив результатов центр контура.

Такой алгоритм применен для того, чтобы в процессе разбить ячейки невыпуклой формы на отдельные ячейки. Результат приведен на рис. 3.

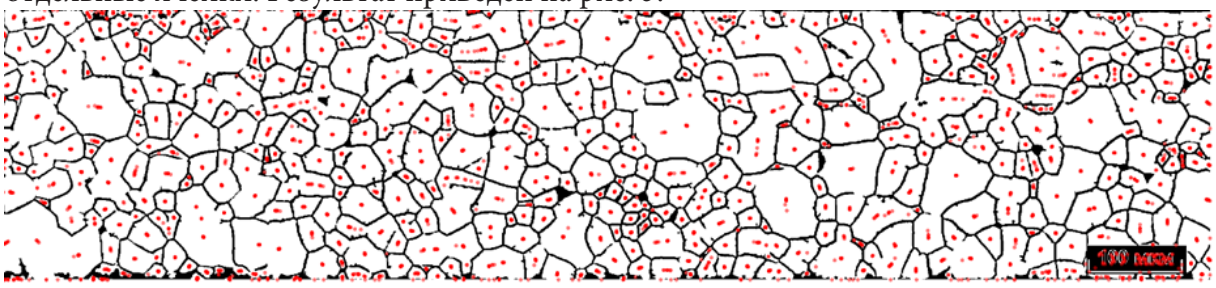


Рис. 3. Первичный результат применения алгоритма, находящего центры ячеек

2.4. Кластеризация

Так как на предыдущем шаге мы получали множество точек почти в одном и том же месте для любого выпуклого контура, их необходимо объединить в одну.

Для этого можно использовать алгоритм кластеризации DBSCAN [6] по нескольким причинам:

– Алгоритм не принимает на вход заранее определенное количество кластеров, а значит, не потребуется проводить предварительный анализ силуэтов или (в нашем случае) как-то заранее подсчитывать ячейки.

– Алгоритм изначально предполагает, что кластеры – сущности большой плотности, разделенные областями малой плотности, а в результате работы предыдущего алгоритма мы имеем именно такую ситуацию.

Однако, для создания объекта DBSCAN необходимо предоставить параметр ε или $min\_samples$, которые отвечают на вопрос, что мы имеем в виду под «большой плотностью». Более формально, кластер формируется, когда существует как минимум $min\_samples$ объектов на расстоянии ε. Кластер строится рекурсивно, рассматривается объект за объектом.

В работе использовано значение $min\_samples$, равное 5, по умолчанию. Параметр ε же подбирался по выборке имеющихся точек: было сформировано распределение расстояний от каждой точки до ее ближайшего соседа, и взят 99-й перцентиль.

После того, как кластеры построены, каждый из них заменен на свою центроиду, и таким образом, количество точек было существенно уменьшено, и каждая из них отвечает за свою ячейку.

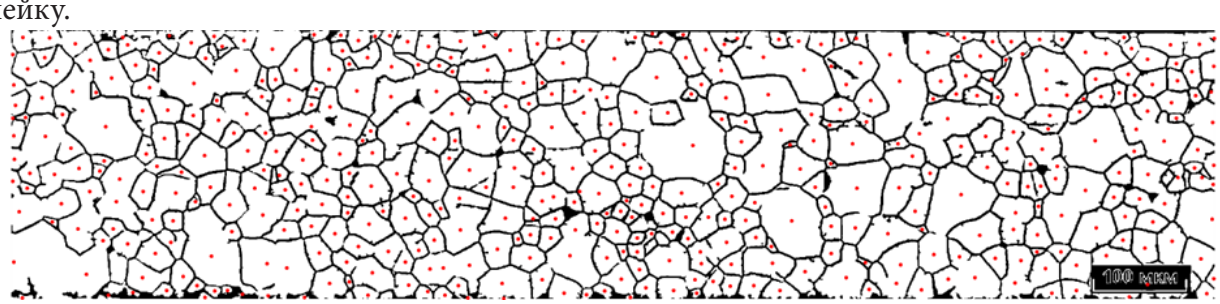


Рис. 4. Результат применения алгоритма DBSCAN

2.5. Алгоритм водораздела

Алгоритм водораздела [7] – классический алгоритм сегментации, то есть для разделения различных областей изображения. Изначально получив набор маркеров, определенных пользователем, алгоритм трактует степень яркости того или иного пикселя как высоту. Далее, алгоритм «заливает бассейны» по маркерам до той степени, пока области разных маркеров не столкнуться друг с другом на одинаковой высоте.

В большинстве случаев маркерами берутся области локальных минимумов изображения. Однако в постановке решаемой задачи отсеянные точки после DBSCAN являются этими маркерами.

2.6. Сравнение распределений

После завершения постобработки мы находим контуры, так же, как и для оригинальной разметки, вычисляем перечисленные метрики и сравниваем распределения. Результат можно увидеть в табл. 1.

Таблица 1

Сравнение подходов по метрике Вассерштейна для ряда метрик

	Площадь	Периметр	Фактор формы	Эквивалентный диаметр
OpenCV-подход	0.0517	0.1056	0.1659	0.1177
U-Net-подход	0.0427	0.0389	0.0857	0.0436

Заключение

Результаты данной работы показывают, что применение сверточных нейронных сетей для решения задач сегментации оправдано: в условиях наличия участков неравномерной контрастности и слабо видимых линий нейросеть может предсказать границу между ячейками.

Литература

1. Wasserstein metric. – Режим доступа: https://en.wikipedia.org/wiki/Wasserstein_metric (Дата обращения: 6.10.2019)

2. Количественный анализ изображений. – Режим доступа: http://structure.by/index.php/metallografiya/31-kolichestvennyj-analiz-izobrazhenij (Дата обращения: 14.03.2020)

3. ГОСТ 5639-82. Стали и сплавы. Методы выявления и определения величины зерна. – Введ. 1983-01-01. – Москва. – 16 с.

4. *Olaf Ronneberger, et al.* U-Net: Convolutional Networks for Biomedical Image Segmentation / Olaf Ronneberger, Philipp Fischer, Thomas Brox // arxiv.org

5. Mean Squared Error. – Режим доступа: https://en.wikipedia.org/wiki/Mean_squared_error (Дата обращения: 11.01.2020)

6. *Daren Wang et al.* DBSCAN: Optimal Rates For Density Based Clustering / Daren Wang, Xinyang Lu, Alessandro Rinaldo // arxiv.org

7. *Jos B.T.M. Roerdink.* The Watershed Transform: Definitions, Algorithms and Parallelization Strategies / Jos B.T.M. Roerdink, Arnold Meijster // Fundamenta Informaticae. – No 41. – P. 187–228.

Жихарева Мария Николаевна – магистрант 2-го года обучения кафедры Математических методов исследования операций факультета Прикладной математики, информатики и механики Воронежского государственного университета. E-mail: zhikharevamarya@yandex.ru

Каширина Ирина Леонидовна – д-р техн. наук, доц., профессор кафедры Математических методов исследования операций факультета Прикладной математики, информатики и механики Воронежского государственного университета. E-mail: kash.irina@mail.ru

УДК 004.932.2

ОБЗОР ЦВЕТОВЫХ МОДЕЛЕЙ БИБЛИОТЕКИ OPENCV

М. В. Зайцев, А. В. Васильев

Воронежский государственный университет

Введение

В научных и коммерческих проектах нередко возникает необходимость взаимодействия с визуальными данными (например, изображениями или видео). Популярный инструмент для работы с ними – библиотека OpenCV, которая позволяет хранить данные об изображении различными способами, в зависимости от выбранной цветовой модели. Описание основных из них и демонстрация их ключевых особенностей на примере реального изображения содержится в данной статье.

OpenCV – кроссплатформенная (Windows, Linux, Mac OS) библиотека компьютерного зрения (процесса преобразования данных, полученных с фотоаппаратов и видеокамер, в новое представление) с открытым исходным кодом, которая разработана на C и C++ с целью повышения вычислительной эффективности [1]. Один из способ цифрового представления полученных данных – двумерная матрица, содержимое элемента которой зависит от выбранной цветовой модели, которые многочисленны, но приблизительны по отношению к тому, как человек воспринимает цвет, поскольку не изучены все значимые аспекты психофизики цветового зрения [2].

Условно модели можно разделить на 4 типа в зависимости от того, как формируются данные:

- RGB-подобные;
- основанные на RGB-подобных(данные получены из RGB-подобных моделей с помощью определённых математических преобразований);
- ориентированные на повторение восприятия цвета человеком;
- экспериментальные модели CIE, основанные на XYZ.

Особенности предварительной обработки изображений зависят от того, какая выбрана модель (то есть сколько компонентов и какая информация содержится в каждом из них). Рассматриваемые модели преимущественно трёхкомпонентные (за исключением, например, однокомпонентной полутоновой; четырёхкомпонентных BGRA и LBGR; пятикомпонентной LBGRA).

1. RGB-подобные модели

Одна из самых популярных таких моделей – двадцатичетырёхбитная **RGB** [3]. Её название – аббревиатура от *Red* (красный), *Green* (зелёный), *Blue* (синий). Есть три числа от **0** до **255** (по **8 бит** – на каждый цвет), они обозначают количество *красного*, *зелёного* и *синего*, соответственно. Итоговый цвет зависит от числового значения каждой из 3-х его составляющих, каналов.

Аналогично устроена и цветовая модель **BGR**, только первое число отвечает за *синий*, а третье – за *красный*, второе всё так же за зелёный. Она по умолчанию используется в OpenCV. Так исторически сложилось из-за востребованности модели BGR в прошлом [4].

Особенности модели BGR позволяют хранимые так данные, полученные из, например, цифрового изображения («Mat inputImage = imread(«Input.png»);») разделить на три

матрицы («vector<Mat> inputImageChannels;») при помощи операции «split(inputImage, inputImageChannels);». На основе полученных данных можно, например, ограничивать информацию по определённому значению содержания красного в итоговом цвете.

В тридцатидвухбитной модели **BGRA** добавляется ещё одно восьмибитное число, отвечающее за **прозрачность** изображения.

В тридцатидвухбитной модели **LBGR** добавляется ещё одно восьмибитное число, отвечающее за **яркость** изображения.

Существует и сорокабитная модель **LBRGA**, в которой есть и число, отвечающее за *яркость*, и число, отвечающее за *прозрачность*.

Но не во всех моделях типа BGR для хранения канала используется 8 бит.

В модели **BGR555** на каждый цвет отводится по *5 бит*. Существуют и модели с другим количеством бит на какой-либо канал, но смысл формирования изображения у них аналогичный.

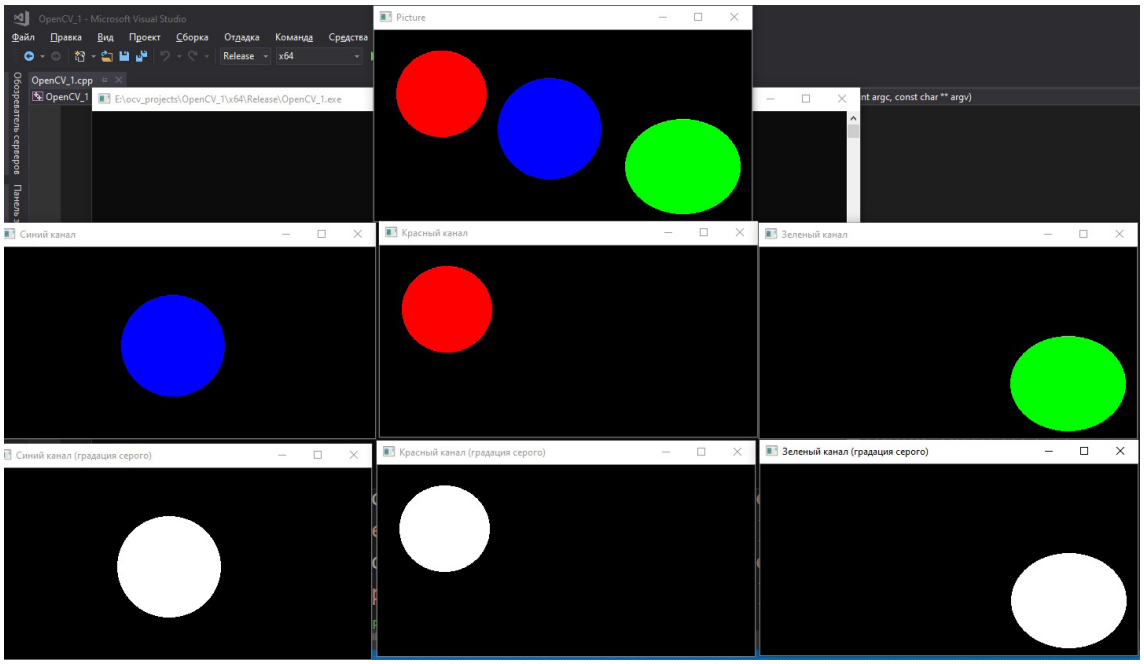


Рис. 1. Визуализация особенностей моделей RGB, BGR

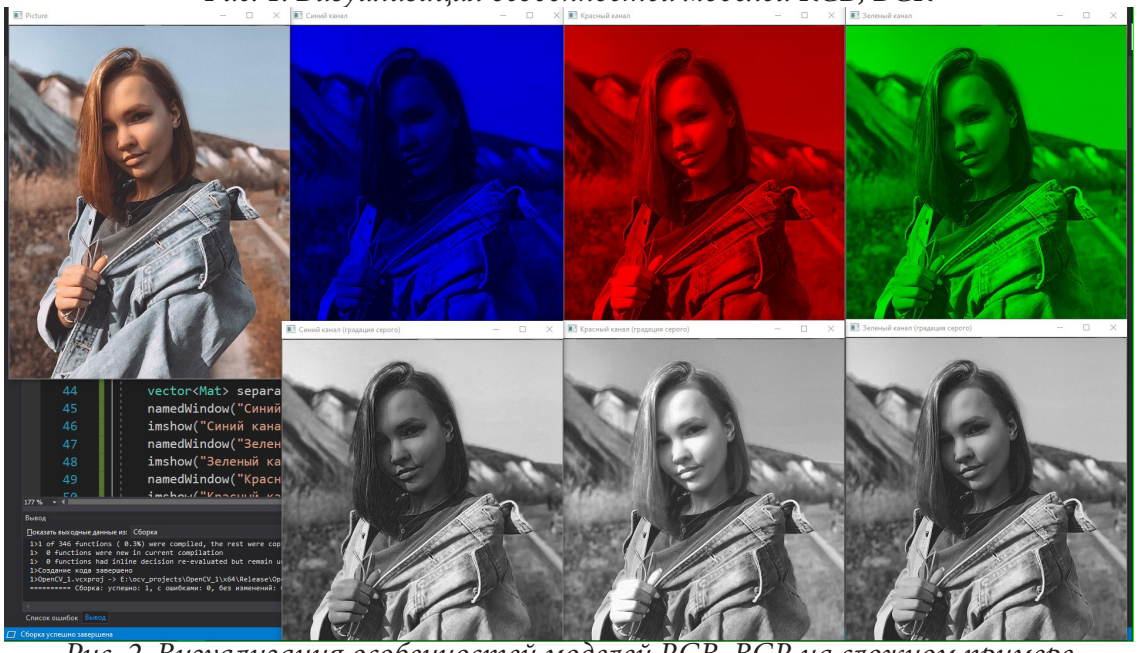


Рис. 2. Визуализация особенностей моделей RGB, BGR на сложном примере

Код для отображения представлен в Листинге 1.

Листинг 1

```
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
using namespace std;
using namespace cv;

vector<Mat> showSepChannels(vector<Mat> channels) {
    vector<Mat>  sepChannels;
    // создание матриц в соответствии с количеством каналов
    for ( int i = 0 ; i < 3 ; i++)
    {
     Mat zeroMatr=Mat::zeros( channels[0].rows, channels[0].cols,channels[0].type());
     vector<Mat> aux;
     for (int j=0; j < 3 ; j++)
     {
            If (j==i)
                  aux.push_back(channels[i]);
            else
                  aux.push_back(zeroMatr);
     }
     Mat  channel;
     merge(aux, channel);
     sepChannels.push_back(channel);
    }
    return  sepChannels;
}
int main(int argc, const char** argv)
{
     //загрузка исходного изображения в матрицу
    Mat pic = imread(«vik1.png»);
    imshow(«Picture», pic);
    vector<Mat> channels;
    split(pic, channels);
    // демонстрация содержимого каналов(градации серого)
    namedWindow(«Синий канал (градация серого)», WINDOW_AUTOSIZE);
    imshow(«Синий канал (градация серого)», channels[0]);
    namedWindow(«Зеленый канал (градация серого)», WINDOW_AUTOSIZE);
    imshow(«Зеленый канал (градация серого)», channels[1]);
    namedWindow(«Красный канал (градация серого)», WINDOW_AUTOSIZE);
    imshow(«Красный канал (градация серого)», channels[2]);
    // демонстрация содержимого каналов(BGR)
    vector<Mat> separatedChannels = showSepChannels(channels);
    namedWindow(«Синий канал», WINDOW_AUTOSIZE);
    imshow(«Синий канал», separatedChannels[0]);
    namedWindow(«Зеленый канал», WINDOW_AUTOSIZE);
    imshow(«Зеленый канал», separatedChannels[1]);
    namedWindow(«Красный канал», WINDOW_AUTOSIZE);
    imshow(«Красный канал», separatedChannels[2]);
    waitKey(0);
    return 0;
}
```

2. Модели, основанные на RGB-подобных

В OpenCV допустимы и модели, которые *получены на основе RGB-подобных*. Например, восьмибитная **полутоновая** модель, в ней каждый пиксель представляется одним числом (*8 бит, информация о яркости*), в самом изображении *присутствуют только оттенки серого цвета*, данные получаются из RGB по следующей формуле [5]:

$$GRAY = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B.$$

Перевести данные из прошлого примера («inputImage») в полутоновое представление в другой матрице («Mat grayImage;») можно при помощи функции **cvtColor** с значением аргумента, отвечающего за то, из какой цветовой модели в какую осуществляется перевод, «*COLOR_BGR2GRAY*»: «cvtColor(inputImage, grayImage COLOR_BGR2GRAY);». Перевод из RGB осуществляется с значением «COLOR_RGB2GRAY» последнего аргумента в функции «cvtColor», из полутонового в BGR и RGB – с значениями «*COLOR_GRAY2BGR*» и «*COLOR_GRAY2RGB*» соответственно. Преобразование из полутоновой модели в RGB осуществляется по формуле:

$$R = GRAY, G = GRAY, B = GRAY.$$

Если применить cvtColor к первому изображению с значением последнего аргумента «COLOR_BGR2GRAY», полученное изображение перевести в третье при помощи функции cvtColor и значения последнего аргумента «COLOR_GRAY2BGR», тогда результат (третье изображение) окажется отличным от изначального (первое изображение) в том случае, если в исходном все элементы матрицы не представляют из себя пиксель значения синего, красного и зелёного которого равны, то есть, за исключением некоторых случаев, данное преобразование ведёт к потерям. Существуют и другие модели, полученные из RGB с помощью определённых преобразований.

Например, **YCrCb**, которая используется в алгоритмах сжатия изображения (*JPEG*) и видео (*MPEG*) [5]. В **Y** хранится информация о яркости пикселя, а в **Cr** и **Cb** – о цвете и насыщенности. Исходя из уравнения, рекомендованного стандартом федеральной комиссии связи:

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$
$$Cr = (R - Y) = R - 0.299 \cdot R - 0.587 \cdot G - 0.114 \cdot B = 0.701 \cdot R - 0.587 \cdot G - 0.114 \cdot B$$
$$Cb = (B - Y) = B - 0.299 \cdot R - 0.587 \cdot G - 0.114 \cdot B = -0.299 \cdot R - 0.587 \cdot G + 0.886 \cdot B,$$

где Cr и Cb при восьмибитном представлении могут принимать значения из диапазона [–128,127]. Перевести изображение из BGR в YCrCb можно с помощью функции cvtColor с значением последнего аргумента «*COLOR_BGR2YCrCb*», наоборот – с «*COLOR_YCrCb2BGR*». По аналогии с RGB изображение в формате «YCrCb» можно разбить на 3 матрицы, содержащие значение каждого отдельного канала (Y, Cr, Cb), и использовать их в дальнейшем анализе изображения.

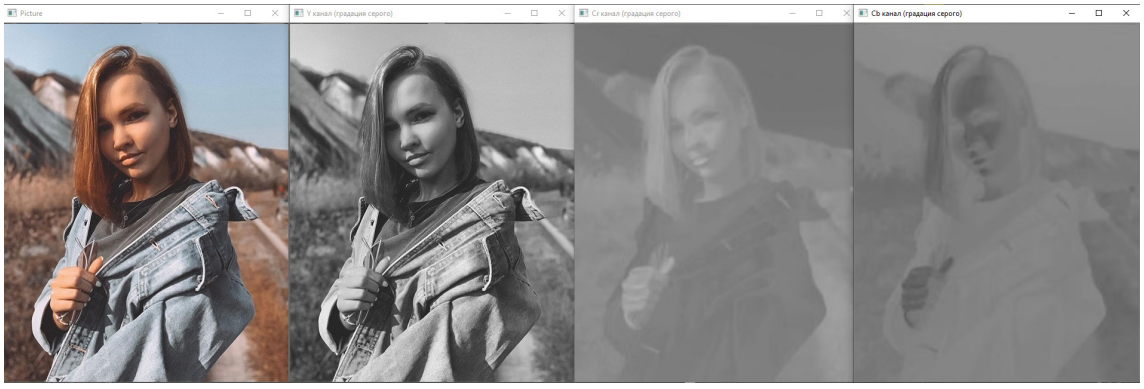


Рис. 3. Визуализация особенностей модели YCrCb

3. Модели, ориентированные на повторение восприятия цвета человеком

Существуют и цветовые модели, которые больше ориентированы на повторение восприятия цвета человеком. Одна из таких – **HSV**. В ней **H** (hue) – *мера спектрального состава* цвета; **S** (saturation) – *насыщенность*, она определяет долю чистого света доминирующей длины волны, то есть чем больше данный параметр, тем «чище» свет, и, наоборот, чем он меньше, тем ближе цвет к серому такой же яркости; **V** (value) – *яркость относительно белого света такой же силы*, описанные каналы соответствуют интуитивно воспринимаемым понятиям тона, насыщенности и светлоты [5].

Цветовую модель HSV можно представить в виде цилиндра (рис. 1) [7].

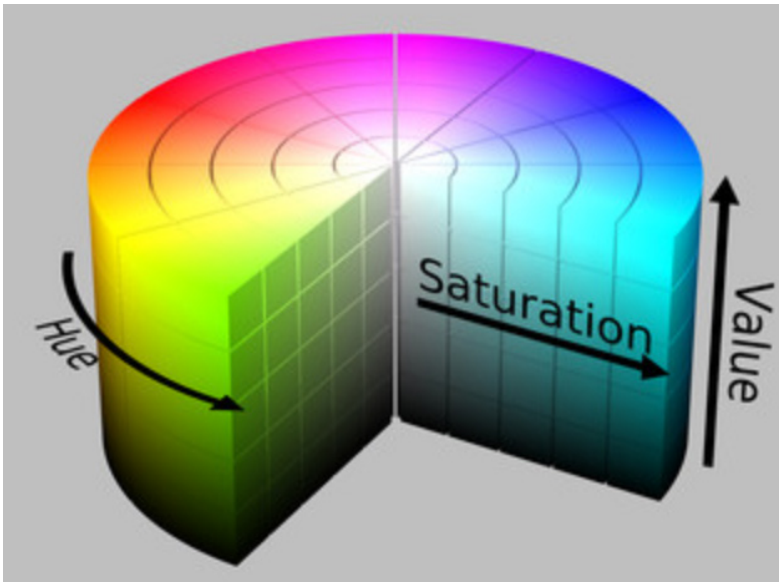


Рис. 4. Графическое отображение модели HSV

Перевести изображение из BGR в HSV с помощью функции cvtColor с последним значением аргумента «*COLOR_BGR2HSV*», наоборот – с «*COLOR_HSV2BGR*».

Если в исходной матрице пиксель описан восьми- или шестнадцатибитным целым, то cvtColor сначала преобразует его в формат с плавающей точкой, приводя значения к диапазону от 0 до 1. Затем преобразование вычисляется по следующим формулам [5]:

$$V = max\left(R, G, B\right)$$

$$S = \begin{cases} \dfrac{V = min\left(R, G, B\right)}{V}, & \text{если } V \neq 0 \\ 0, & \text{иначе} \end{cases}$$

$$H = \begin{cases} \dfrac{60\left(G - B\right)}{V - min\left(R, G, B\right)}, & \text{если } V = R \\ 120 + \dfrac{60\left(B - R\right)}{V - min\left(R, G, B\right)}, & \text{если } V = G \\ 240 + \dfrac{60\left(R - G\right)}{V - min\left(R, G, B\right)}, & \text{если } V = B \end{cases}$$

Если $H < 0,$ то $H = H + 360.$ В конце значения преобразуются обратно к требуемому типу данных.

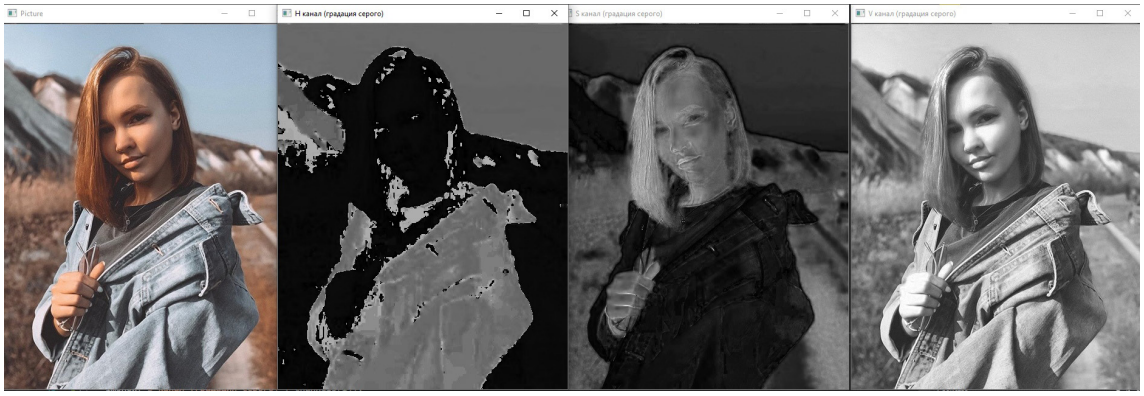


Рис. 5. Визуализация особенностей модели HSV

Цветовая модель **HLS** выстроена похожим образом, что и HSV. В HLS цвет представлен *оттенком*, *светлотой* и *насыщенностью*. Но светлота чистого цвета, определенная в HLS, равна светлоте среднего серого, тогда как яркость чистого цвета, определенная в HSV, равна яркости белого[5].

Перевести изображение из BGR в HLS можно с помощью функции cvtColor с значением последнего аргумента «*COLOR_BGR2HLS*», наоборот – с «*COLOR_HLS2BGR*».

Как и в случае HSV, если пиксели входного изображения представлены восьми- или шестнадцатибитным целым числом, то cvtColor сначала преобразует его в формат с плавающей точкой, приводя значения к диапазону от 0 до 1, затем преобразование вычисляется по следующим формулам [5].

$$Vmax = max\left(R, G, B\right)$$

$$Vmin = min\left(R, G, B\right)$$

$$L = \frac{Vmax + Vmin}{2}$$

$$S = \begin{cases} \dfrac{Vmax - Vmin}{Vmax + Vmin}, & \text{если} \quad L < 0.5 \\ \dfrac{Vmax - Vmin}{2 - \left(Vmax + Vmin\right)}, & \text{если} \quad L \geq 0.5 \end{cases}$$

$$H = \begin{cases} \dfrac{60\left(G - B\right)}{S}, & \text{если} \quad Vmax = R \\ 120 + \dfrac{60\left(B - R\right)}{S}, & \text{если} \quad Vmax = G \\ 240 + \dfrac{60\left(R - G\right)}{S}, & \text{если} \quad Vmax = B \end{cases}$$

Если $H < 0,$ то $H = H + 360.$ В конце значения преобразуются обратно к требуемому типу данных.

4. Экспериментальные модели CIE, основанные на XYZ

Существуют и экспериментально полученные модели. Например, XYZ. В ней цвет описывается фотометрической яркостью излучения Y, которая связана с чувствительностью глаза к яркости света и двумя дополнительными каналами X и Z, стандартизованными международной комиссией по освещению (CIE) [5].

Рис. 6. Визуализация особенностей модели HLS

Перевести изображение из BGR в XYZ можно с помощью функции cvtColor с значением последнего аргумента «*COLOR_BGR2XYZ*», наоборот – с «*COLOR_XYZ2BGR*».

Преобразование осуществляется с помощью следующих формул [5]:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.375780 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.240479 & -1.53715 & -0.498535 \\ -0.969256 & 1.875991 & 0.041556 \\ 0.055648 & -0.204043 & 1.057311 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

В XYZ цвета масштабируются неодинаково [5], потому у её разработчиков, CIE, возникла необходимость в новых моделях, которыми стали **Lab** и **Luv**.

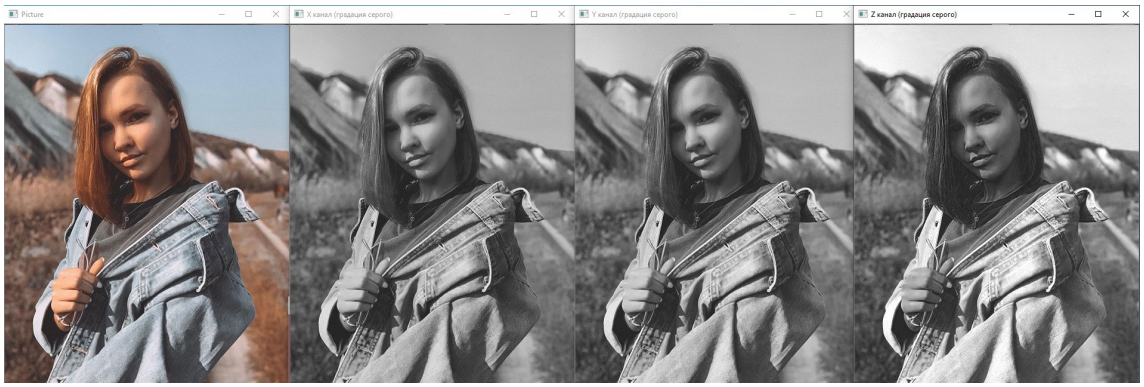


Рис. 7. Визуализация особенностей модели XYZ

В Lab **L** определяет *яркость цвета*, **a** – *соотношение красного и зелёного*, **b** – *соотношение синего и зелёного*, из XYZ получается Lab по следующим формулам[3]:

$$L = \begin{cases} 116\left(\frac{Y}{Y_0}\right)^{\frac{1}{3}} - 16, & \text{если } \frac{Y}{Y_0} > 0.008856 \\ 903.3\frac{Y}{Y_0}, & \text{иначе} \end{cases}$$

$$a = 250 \cdot \left(f\left(\left(\frac{X}{X_0}\right)^{\frac{1}{3}}\right) - f\left(\left(\frac{Y}{Y_0}\right)^{\frac{1}{3}}\right) \right)$$

$$b = 100 \cdot \left(f\left(\left(\frac{Y}{Y_0} \right)^{\frac{1}{3}} \right) - f\left(\left(\frac{Z}{Z_0} \right)^{\frac{1}{3}} \right) \right)$$

где $f\left(t^{\frac{1}{3}}\right) = \begin{cases} t^{\frac{1}{3}}, & \text{если } t > 0.008856 \\ 7.787 \cdot t + \frac{16}{116}, & \text{иначе} \end{cases}$

X_0, Y_0, Z_0 – координаты опорного белого цвета в системе XYZ.

Перевести изображение из BGR в Lab можно с помощью функции cvtColor с значением последнего аргумента «*COLOR_BGR2Lab*», наоборот – с «*COLOR_Lab2BGR*».

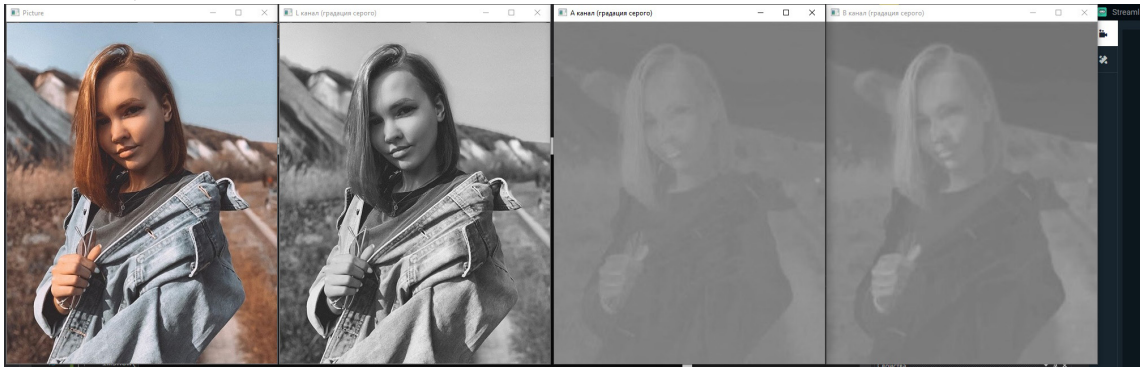


Рис. 8. Визуализация особенностей модели Lab

В Luv значения L, u и v получаются по следующим формулам [8]:

$$u' = \frac{4X}{X + 15Y + 3Z} = \frac{4x}{-2x + 12y + 3}$$

$$v' = \frac{9Y}{X + 15Y + 3Z} = \frac{9y}{-2x + 12y + 3}$$

$$L = \begin{cases} \left(\frac{29}{3}\right)^3 \cdot \frac{Y}{Y_n}, & \text{если } \frac{Y}{Y_n} \le \left(\frac{6}{29}\right)^3 \\ 116 \cdot \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16, & \text{если } \frac{Y}{Y_n} > \left(\frac{6}{29}\right)^3 \end{cases}$$

$$u^* = 13L \cdot \left(u' - u'_n\right)$$

$$v^* = 13L \cdot \left(v' - v'_n\right).$$

Перевести изображение из BGR в Luv можно с помощью функции cvtColor с значением последнего аргумента «*COLOR_BGR2Luv*», наоборот – с «*COLOR_Luv2BGR*».

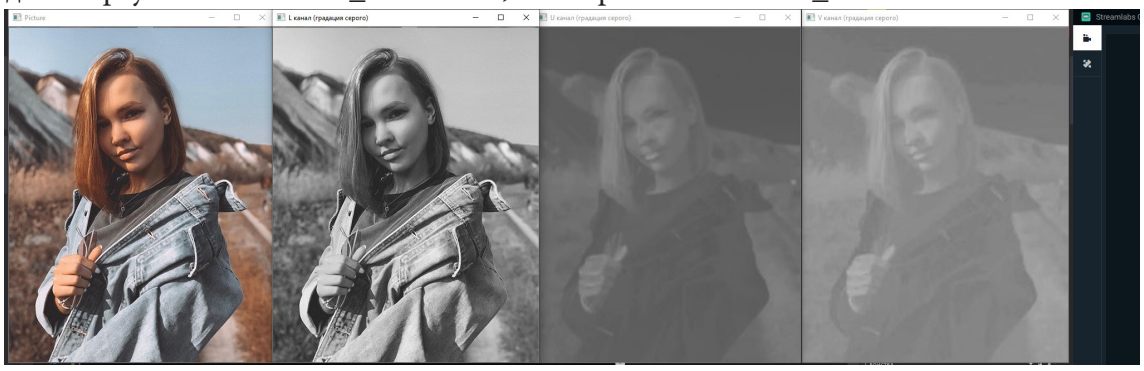


Рис. 9. Визуализация особенностей модели Luv

Заключение

Рассмотренные цветовые модели необходимы для различных задач. В зависимости от того, что нужно анализировать в конкретном случае: значение красного, насыщенность или другой параметр. Основные цветовые модели подходят для решения задач, связанных с компьютерным зрением. Выбор цветовой модели может оказать значительное влияние на дальнейший алгоритм предварительной обработки изображения.

Литература

1. *Bradsky G., Kaehler A.* Learning OpenCV. O'Reilly, 2008. – 571 p.
2. *Марр Д.* Зрение. Информационный подход к изучению представления и обработки зрительных образов. – М. : Радио и связь, 1987.
3. *Фисенко В. Т., Фисенко Т. Ю.* Компьютерная обработка и распознавание изображений: учеб. пособие. – СПб. : СПбГУ ИТМО, 2008. – 192 с.
4. Learn OpenCV. – https://www.learnopencv.com/color-spaces-in-opencv-cpp-python/ (дата обращения: 23.04.20).
5. *Глория Буэно Гарсия, Оскар Дениз Суарес, Хосе Луис Эспиноса Аранда, Хесус Салидо Терсеро, Исмаэль Серрано Грасиа, Ноэлия Валлез Энано* Обработка изображений с помощью OpenCV / пер. с англ. Слинкин А. А. – М. : ДМК Пресс, 2016. – 210 с.
6. *Быков Р. Е.* Основы телевидения и видеотехники Горячая линия - телеком, 2006. 399 с.
7. OpenCV Open Source Computer Vision. – https://docs.opencv.org/trunk/da/d97/tutorial_threshold_inRange.html (дата обращения: 23.04.20).
8. Wikipedia. – https://en.wikipedia.org/wiki/CIELUV (Дата обращения: 23.04.20).

Зайцев Михаил Владимирович – магистрант 2-го года обучения кафедры математических методов исследования операций Воронежского государственного университета.
E-mail: zajcev.mikhail@gmail.com

Васильев Александр Владимирович – магистрант 1-го года обучения кафедры онтологии и теории познания Воронежского государственного университета.
E-mail: alexandecsvsu@gmail.com

УДК 004.042

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ПОД ПЛАТФОРМУ IOS ДЛЯ ПОМОЩИ ОРГАНИЗАЦИИ МЕРОПРИЯТИЙ

Ю. Н. Затонская

Воронежский государственный университет

Введение

Современный мир можно охарактеризовать стремительным расширением информационных технологий, в связи с чем возрастает популярность использования мобильных устройств в повседневной жизни. Всего пару лет назад данные девайсы находили своё применение в основном в руках населения в возрасте до 45 лет, сейчас же подобного деления пользователей по возрастной категории просто не существует.

С развитием технологий растут и потребности общества в познании: появляются различные конференции и форумы, увеличивается количество тематических фестивалей с целью ознакомления потребителя с новыми продуктами. С ростом количества мероприятий по всему миру повышается потребность в продуктах, способных облегчить и ускорить организацию, что приводит к необходимости появления специальных мобильных помощников.

Приложение для организации мероприятий является сервисом, позволяющим облегчить и систематизировать основные аспекты планирования, возникающие в процессе разработки и оформления различных встреч, форумов или конференций. В подобном программном обеспечении нуждаются специалисты в сфере event-менеджмента, маркетинга и пиара. Ежегодно более половины маркетологов, организующих мероприятия, используют специальные мобильные приложения с целью не только значительно сэкономить время, но и упростить участникам взаимодействие с организатором. Данный подход заметно повышает качество обслуживания, что приводит к привлечению новой аудитории и оптимизации усилий.

Для удовлетворения потребностей и развития современного общества использование информационных технологий крайне необходимо. А в связи с тем, что с организацией мероприятий сталкиваются во всех сферах, от частных развлекательных фестивалей до крупных научных конференций, повышается и актуальность создания данного приложения как для разработчика, так и для пользователя.

Разработка и анализ приложения

Написание мобильного приложение под платформу IOS предполагает, что в основе программы лежит пакет инструментов Xcode. В качестве языка программирования используется Swift. Для разработки интерфейса Xcode предоставляет следующие методологии: storyboard, набор xib-фалов. В зависимости от целей предоставленные ресурсы можно использовать как вместе, так и по отдельности. В связи с большим набором используемых экранов, количество и деятельность которых отражены на рис.1, было решено использование одновременно xib и storyboard, что предоставило возможность создавать дизайн приложения наглядно, как через Interface Builder, и вручную через код, а также позволило избежать одного перегруженного ссылками файла с интерфейсом.

Приложение предоставляет пользователю две возможные роли: организатор или участник, которая определяется при входе. Это влияет на предоставленные действия.

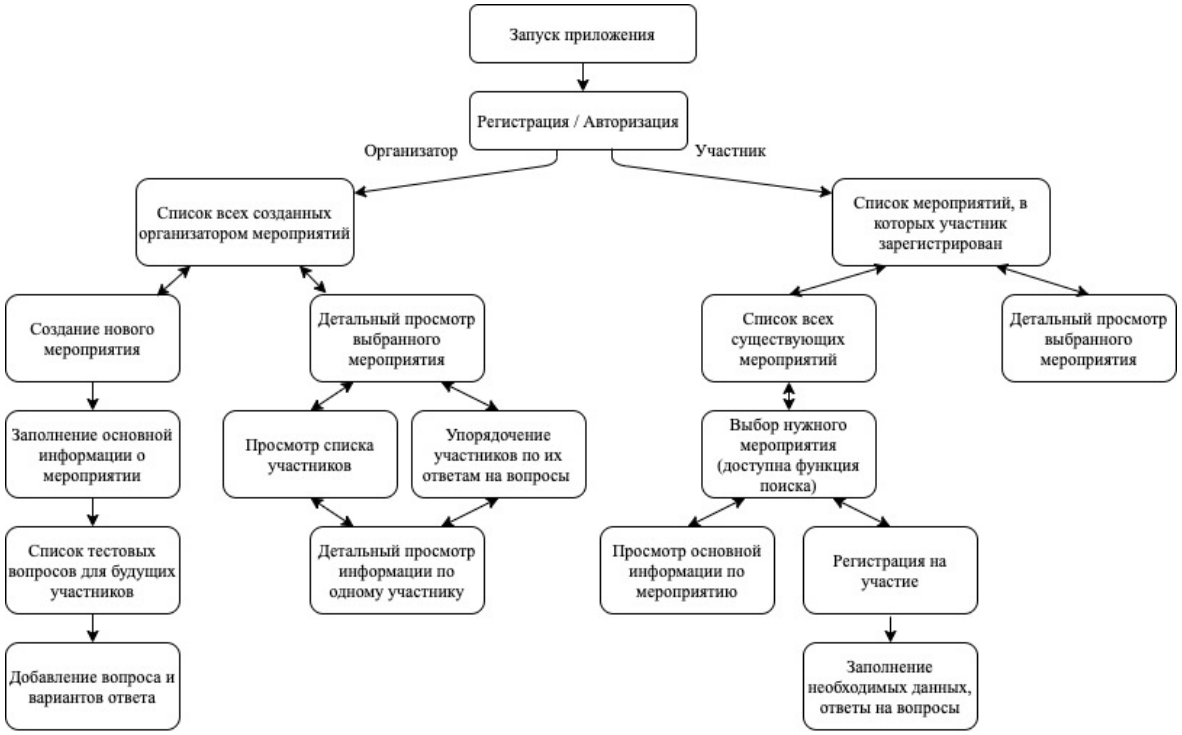


Рис.1. Структура приложения

В процессе разработки были созданы соответствующие классы. Основными являются:

– ViewController() – базовый класс, отображающий первую страницу приложения;

– EventsViewController() – используется участником, содержит список всех существующих мероприятий с возможностью поиска по словам. Поиск организован зависимостью от UISearchBarDelegate;

– ThemesViewController() – используется организатором, отображает список всех созданных им мероприятий;

– CreateMeetingViewController() – отвечает за одну из главных задач всего приложения: создание мероприятия;

– CreateQuestionViewController() – в данном классе организатором создается список вопросов для участника, чтобы в дальнейшем, проанализировав все ответы, правильно организовать взаимодействие гостей.

Основную часть пользовательского интерфейса занимает табличное представление. UITableView организует данные в виде списка с прокруткой множества строк, для каждой из которых предоставлена возможность детального просмотра. Такой подход выбран с целью обеспечения удобства просмотра созданных мероприятий. Практически каждый класс представлен описанным способом.

Для хранения данных о пользователях используется база данных Firebase, которая изменяется в реальном времени и хранит данные в формате JSON. Любые изменения в базе данных тут же синхронизируются.

Анализ концепций конкурирующих продуктов и отзывов пользователей помог составить список необходимых составляющих приложения для создания актуального и удобного сервиса.

Пожелания организаторов:

– оптимизация регистрации и опроса участников;

– получение списка гостей;

– доступ к информации о каждом посетителе;

– ведение нескольких мероприятий одновременно.

Пожелания участников:
– легкий поиск необходимого мероприятия;
– актуальное расписание;
– оповещения о важных событиях или изменениях;
– возможность связаться с организатором.

Заключение

Результаты исследования изложенной темы востребованы в вопросах использования приложений мобильных телефонов в event-индустрии. Сделан вывод, что применение специальных программных обеспечений сокращает шаблонные действия и оптимизирует время работы организатора, способствует повышению комфорта и вовлеченности участника, поднимает престиж и конкурентоспособность компании.

Данная тема не только не утратит свою значимость, но и обязательно будет приобретать особую ценность с течением времени, ведь развитие информационных технологий влечет за собой потребность общества в адаптации этих новшеств к жизни. Уже на текущем этапе внедрение мобильных приложений для организации мероприятий в работу значительно повышает статус предприятий, которые их используют, структурирует и оптимизирует деятельность организатора и позволяет участнику легче ориентироваться в происходящем. Задача разработчика: создавать качественный продукт, соответствующий спросу.

В процессе исследования и разработки выполнено:

1. Анализ актуальности выбранной тематики с детальным обзором существующих продуктов и отзывов по ним.

2. Изучение подходов event-менеджеров к работе через литературные источники.

3. Создание мобильного приложения, содержащего необходимые для функционирования пункты.

4. Детальная проработка интерфейса программы с целью создания привлекательного для пользователя продукта.

В будущем планируется доработать мобильное приложение, добавив возможность регистрации через социальные сети, общий чат всех гостей, синхронизацию с календарем телефона и возможность покупки билетов на мероприятие.

Литература

1. Matthew Knott. Beginning Xcode: Swift Edition. – Apress, 2014. – 75 с.

2. Paris Buttfield-Addison, Jon Manning, Tim Nugent. Learning Swift: Building Apps for macOS, iOS, and Beyond (2nd Edition). – O'Reilly Media, April 27, 2017. – 170 с.

3. FireBase. Интернет-энциклопедия. – URL: https:// ru.wikipedia.org/wiki/Firebase (дата обращения: 20.03.2020)

4. Введение в Firebase: пишем простое социальное приложение на Swift. – URL: https://habr.com/ru/post/277941/(дата обращения: 20.03.2020)

5. Официальный сайт Firebase. – URL: https:// firebase.google.com/ (дата обращения: 22.03.2020)

Затонская Юлия Николаевна – студентка 4-го курса кафедры МО ЭВМ Воронежского государственного университета. E-mail: zatonskayay@gmail.com

Болотова Светлана Юрьевна (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры МО ЭВМ Воронежского государственного университета. E-mail: Bolotova.svetlana@gmail.com

УДК 004.622

РАЗРАБОТКА АВТОМАТИЗИРОВАННОГО РАБОЧЕГО МЕСТА КАССИРА

А. А. Капранчикова, М. П. Лукин

Воронежский государственный университет

Введение

В современном мире широкое распространяется и быстро развивается направление автоматизации бизнес-процессов. Это позволяет тратить намного меньше времени на рутинные занятия и увеличивает скорость выполнения задач. Именно поэтому автоматизация процесса продажи очень важно.

Автоматизированная билетная касса кинотеатра помогает оптимизировать процесс реализации билетов, практически избавиться от очередей, существенно сократив время на обслуживание одного клиента, а также использовать разные каналы продажи билетов. Билетная система для театра, цирка, ледового дворца, стадиона и других зрелищных залов абсолютно идентична.

Также автоматизированная билетная касса кинотеатра очень облегчает работу бухгалтерии и менеджеров репертуарного плана, в автоматическом режиме генерируя отчетность за любой период, по любым указанным параметрам.

В данной работе описывается процесс проектирования и разработки web-приложения, предназначенного для автоматизации деятельности билетных касс.

Цель данной работы написать удобное и универсальное приложение для продажи билетов на различные мероприятия и события.

1. Анализ

1.1. Общий анализ

Для решения поставленной задачи необходимо разработать пакет инструментов, позволяющий выполнять действия с билетами и заказами. Приложение должно быть разбито на логические модули. Также понадобятся вспомогательные библиотеки для удобства написания кода.

Общий вид взаимодействия пользователя с приложение представлен на рис. 1.1.

1.2. Анализ существующих решений

В России существует не так уж и много билетных систем, есть несколько компаний, которые еще с конца прошлого века – начала 21-го века разрабатывают свои решения. Набрав базу клиентов, они также успели подстроиться под развитие онлайн-продаж и разработали веб-системы продаж билетов.

TicketSoft, TicketNet и Супербилет создали и поддерживают билетные системы для кинотеатров, театров, стадионов и других площадок. Они содержат большой услуг и функций. Не только продажа билетов, но и учет билетных катушек и подсчет внесенных наличных за смену. Билетные системы также включают в себя пропускные системы, кассы и кассовые аппараты для самообслуживания и др.

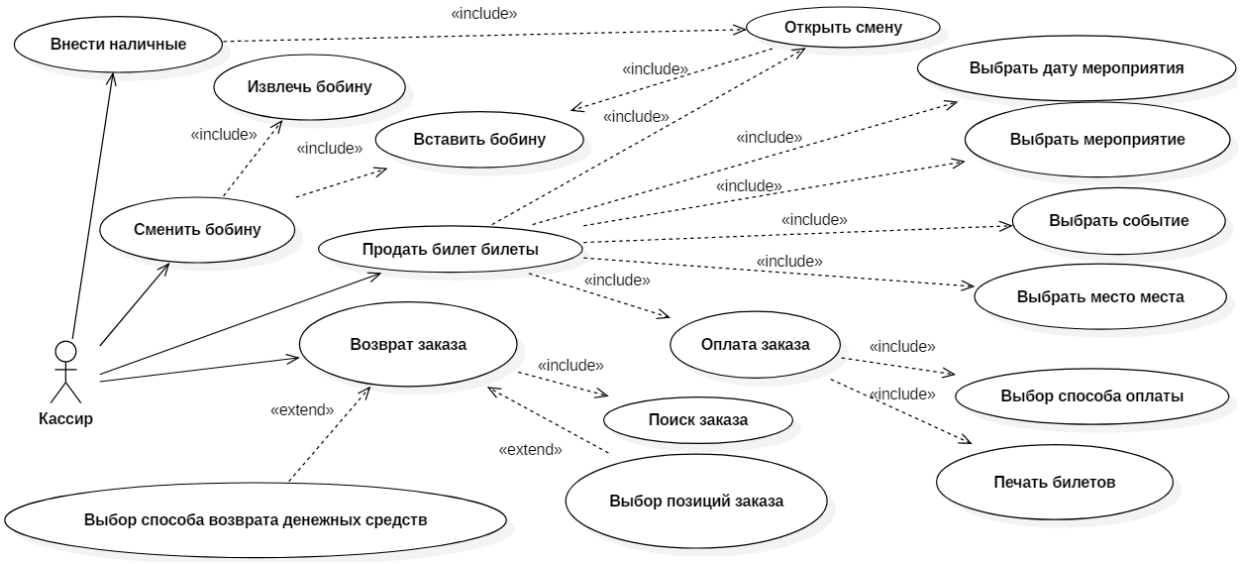


Рис. 1.1. Диаграмма вариантов использования

Для каждой билетной системы, можно указать для чего она лучше подходит, так, например, билетная система TicketSoft предназначается для спортивных мероприятий, для музеев и кино. TicketNet ориентирована на музеи, а Супербилет подойдет для музеев, театров, спортивных мероприятий и кинотеатров.

У всех систем есть один общий минус. Он заключается в том, что работа администратора и кассира не разделены, а представлены в общем интерфейсе.

2. Реализация

2.1. Структура данных

Проект состоит из большого количества компонентов. Каждый компонент включает в себя файл шаблона, файл стилей и файл с логикой. Компоненты позволяю логически разбить клиентскую часть на разные блоки. для обращения к базе данных используются сервисы, которые обращаются к базе данных и возвращают Observable, с которым уже работают нужные компоненты.

Общая схема приложения представлена на рис. 2.1. Пользователь обращается к Angular приложению, когда обращается к сайту через конкретную ссылку. Приложение в свою очередь начинает загрузку компонентов сверху вниз, то есть изначально загружается главный компонент AppComponent, затем благодаря роутингу мы понимаем какой компонент загружать дальше. На нашем рисунке видно, что загружаемый компонент имеет название EventComponent. Дальше загружаются компоненты, которые включены в EventComponent, а именно, HeaderComponent и EventList, первый отвечает за отображение шапки сайта, а второй за список мероприятий.

Для того, чтобы отобразить список мероприятий приложение должно обратиться к нашему серверу, а делается это с помощью сервиса, который так же прописан в Angular приложении и общается к серверу через http запросы. Полученные мероприятия отправляются в компонент и уже компонент понимает, как следует отображать, полученные данные.

Общая схема прикладной архитектуры билетной системы представлена на рис. 2.2. АРМ Кассира обращается к серверу, который выполняет такие функции как:

• аналитика – предназначена для анализа данных о посещении. Подсистема обеспечивает автоматизацию процессов сбора, поиска и статистической обработки информации;

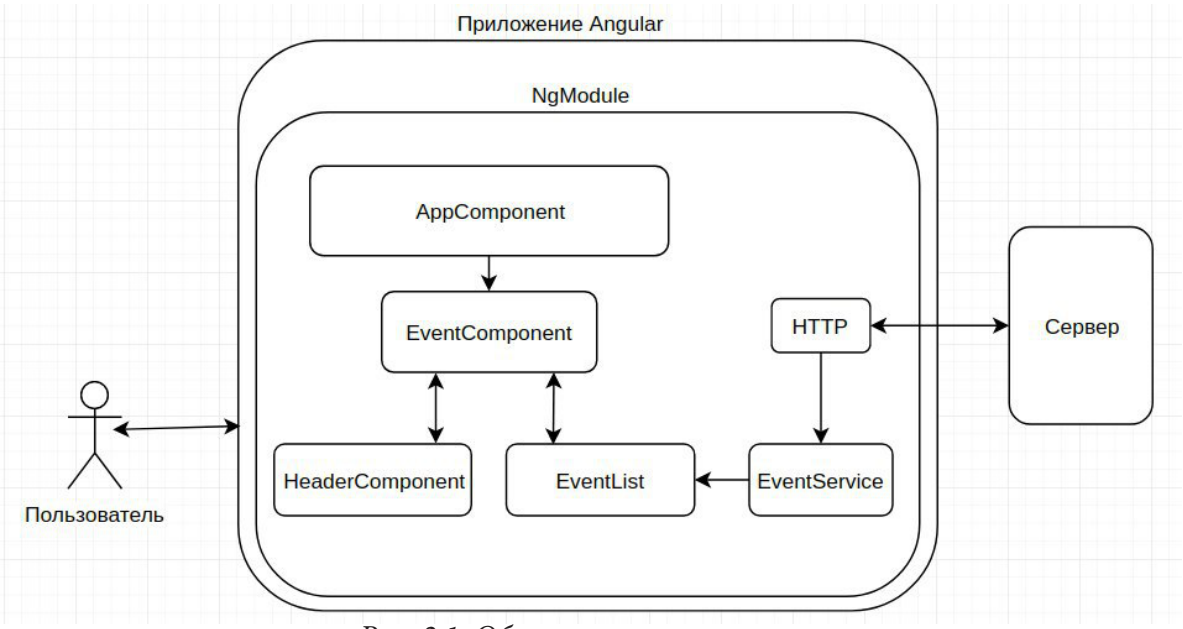


Рис. 2.1. Общая схема приложения

• управление площадкой – подсистема предназначена для управления и контроля за деятельностью площадки в рамках реализации билетов, абонементов и экскурсий, а также обеспечения учета и контроля посетителей;

• контроль билетов – подсистема предназначена для обеспечения процессов контроля и учета билетов на территории учреждения;

• интеграция – подсистема предназначена для обеспечения процессов взаимодействия разрабатываемой системы с оборудованием и смежными системами;

• НСИ – подсистема предназначена для конфигурирования политик разграничения прав доступа пользователей подсистемам и функциям Системы;

• публикация событий – подсистема предназначена обеспечения возможности публикации музейных событий на официальном сайте Мэра Москвы.

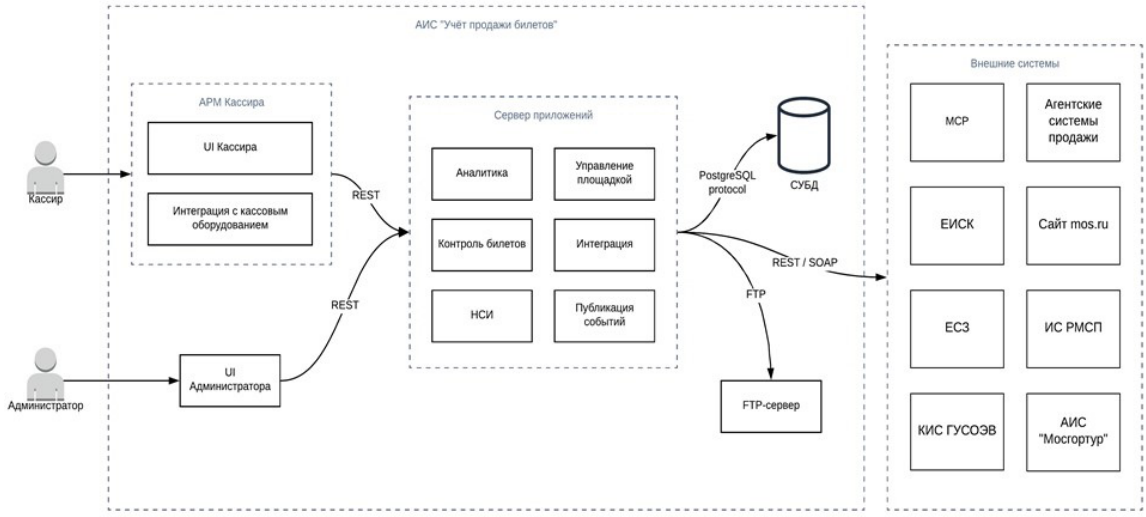


Рис. 2.2. Схема прикладной архитектуры

2.2. Маршрутизация

Для работы маршрутизации в Angular проекте очень важным является модуль RouterModule, который хранится в npm пакете @angular/router. Поэтому если вы хотите добавить маршрутизацию в ваше приложение, вы должны добавить этот модуль в список зависимостей своего проекта, а именно в файл, который располагается в корне приложения и называется package.json.

Для того чтобы указать всевозможные маршруты в проекте должен быть создан специальный файл, который называется app-routing.js. В этом файле определены все маршруты или по-другому все ссылки на страницы приложения. Для добавления маршрута к приложению необходимо в переменную со всеми маршрутами, который представляет из себя массив из объектов, добавить еще один объект, в котором необходимо указать поля path – путь, по которому будет открываться указанный далее компонент, и сам компонент component, при необходимости можно указать data–данные, которые можно будет получить в скрипте компонента, а также можно указать child для вложенного маршрута. Также при необходимости можно указать такие параметры как canActivate и canDeactivate, которые нужны для того, чтобы добавить логики в моменты перехода на определенную страницу и уход с нее.

Заключение

В результате проделанной работы было успешно реализовано и протестировано web-приложение для автоматизации рабочего места кассира, функциональные возможности которого включают:

- авторизация в системе;
- открытие смены;
- указание внесенных наличных;
- указание вставленной билетной катушки;
- продажа билетов на мероприятия со схемой зала;
- продажа билетов на мероприятия без схемы зала;
- проход школьников по карте «Москвенок»;
- бронирование билетов;
- возврат билетов;
- оплата забронированных билетов;
- печать Х-отчетов;
- печать отчетов по заказам;
- закрытие смены.

Данное приложение рассчитано для упрощения продажи билетов на различные мероприятия.

На данный момент автоматизированное рабочее место кассира внедряется в музей Москвы. Началась разработка автоматизированного рабочего места администратора для настройки и управления мероприятиями и событиями, которые отображаются в системе кассира.

В дальнейшем работа над приложением будет продолжена, планируется вести поддержку автоматизированного места кассира, настроить оплату через банковские карты, а также продолжить реализовывать автоматизированное рабочее место администратора.

Литература

1. Tutorials and references relating to HTML, CSS, JavaScript, JSON. – Режим доступа: https://www.w3schools.com

2. Онлайн учебник javascript. – Режим доступа: https://learn.javascript.ru

3. Справочник HTML, CSS. – Режим доступа: http://htmlbook.ru

4. MDNwebdocs. – Режим доступа: https://developer.mozilla.org
5. GitLabDocs. – Режим доступа: https://docs.gitlab.com
6. Федеральный закон РФ «Об информации, информационных технологиях и о защите информации» от 27 июля 2006 г. N 149-ФЗ.
7. ГОСТ 34.602-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплексность и обозначение документов при создании автоматизированных систем. – Взамен ГОСТ 24.201-85; введ. 01.01.90.
8. ГОСТ 34.601-90 Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания. – Взамен ГОСТ 24.601-86; введ. 01.01.92.
9. AngularDocs. – Режим доступа: https://angular.io/docs
10. Angular Material. – Режим доступа: https://material.io/
11. RxJSDocs .– Режим доступа: https://www.learnrxjs.io/
12. MomentsJSDocs. – Режим доступа: https://momentjs.com/docs/
13. Lodash. – Режим доступа: https://lodash.com/

Капранчикова Алисия Александровна – магистрант 1-го года обучения кафедры Программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: kapranchikova_al@mail.ru

Лукин Марк Петрович – магистрант 1-го года обучения кафедры Программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: mark.lukin36@gmail.com

Барановский Евгений Сергеевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: esbaranovskii@gmail.com

УДК 004.052.32

ЗАЩИЩЕННАЯ МОДЕЛЬ КОРПОРАТИВНОГО ЦЕНТРА ОБРАБОТКИ ДАННЫХ

А. Н. Киселева

Воронежский государственный университет

Введение

Центр обработки данных (ЦОД) – это комплекс мощных серверов, дисковых хранилищ и технических решений, предназначенных для автоматизации и бесперебойной работы коммерческих процессов[1].

Рис. 1. Типовая схема ЦОД

Центры обработки данных в основном делят на корпоративные и коммерческие.

Для обеспечения функционирования виртуальных сервисов и хранения информации компании, используют корпоративные[1].

А для хранения и обработки данных сторонних пользователей в целях повышения эффективности экономической деятельности, необходимы коммерческие.

При этом основном целью и задачей остается создание благоприятных и защищенных условий для доступа конкретной компании к собственным данным и их закрытию от посторонних пользователей. Достигается это с помощью:

- Хранения и анализа больших объемов информации;
- Обеспечения безопасности и безотказности высокотехнологичных систем;
- Обеспечения максимальной доступности данных;
- Объединения отдельных составляющих ИТ-систем.

1. Задачи при защите ЦОД

Важной задачей считается решение проблем защищенности и отказоустойчивости систем. Полноценная защита центра обработки данных предполагает использование интегрированной многоуровневой системы обеспечения информационной безопасности .

Рассмотрим три основных:

К первому уровню относится разделение на макро- и микросегменты, охватывающие множество функциональных направления, сервисы и группы пользователей из числа бизнеса.

Второй связан с выявлением аномалий внутри сегментов, обеспечением безопасности на уровне гипервизора. Трафик здесь анализируется с использованием продвинутых алгоритмов поиска, способных распознать аномальные активности, несущие вред.

Третий, заключительный, связан с защитой оконечных устройств (серверов, виртуальных машин) с помощью способа использующего агента или нет.

Для большинства дата-центров эти способы существенно повышают степень информационной безопасности. Обеспечивая всестороннюю безопасность ЦОД, требуется система эффективная в пяти основных областях, для решения следующих задач:

- адаптироваться к изменениям;
- охватывать весь период атаки;
- обеспечивать мониторинг и контроль разработанных на заказ приложений ЦОД;
- обрабатывать асимметричные потоки трафика и транзакции приложений между устройствами или центрами обработки данных;
- защищать сеть целиком.

2. Основные подходы к созданию ЦОД

Приступая к созданию центра обработки данных, необходимо определить подход, который мы будем использовать. Выделим и рассмотрим три основных.

Первый и ожидаемый – это традиционный, подразумевается строительство соответствующего требованиям заказчика ЦОД в специально выделенном помещении. Применение стандартных апробированных элементов гарантирует их совместимость друг с другом, обеспечивает высокий уровень надежности, сокращает время, необходимое для разработки и реализации решения.

Так же можно выбрать вариант мобильного ЦОД. Это достаточно молодое решение, еще не набравшее популярности. Основные преимущества этого подхода – возможность предельно быстрого развертывания корпоративной ИТ-инфраструктуры, быстрый ввод дополнительных мощностей, а также возможность размещения ЦОД в необорудованном помещении.

Так же есть вариант совсем отказаться от постройки центра обработки данных и арендовать его. Популярность услуги обуславливается доступностью и удобством данного подхода для заказчика. У данного подхода есть и слабые стороны: многие компании не готовы отдавать на сторону хранение и обработку своих данных, и вынуждены строить собственный ЦОД, чтобы иметь полный контроль с соблюдением конфиденциальности.

3. Проблематика построения центров обработки данных

Как и в любой сфере здесь есть свои проблемные места. Затрудненным моментом является целостность системы, состоящая из взаимосвязанных программных и аппаратных компонентов.

Основные вопросы связаны с доступностью, защитой и хранением данных. Необходимо обеспечивать и следить за:

- отказоустойчивостью;
- катастрофоустойчивостью;
- защищенностью данных;
- интеллектуальным хранением данных.

Инфраструктуре постоянно необходимо адаптироваться под запросы бизнеса. Изменения должны быть динамичными и не должны нести задержки и остановку бизнес-процесса.

Решение данных вопросов в рамках ЦОД позволяет улучшить результат при минимальных затратах, поэтому можно с уверенность сказать о более эффективном использовании имеющихся ресурсов.

Долгое время основу центра обработки данных составляли в основном большие и дорогие серверные решения, на данный момент численность бюджетных систем возросло. Необходимо отметить, что большое значение для ЦОД несут системы хранения данных и средства резервного копирования.

Заключение

Неоспоримым является актуальность и необходимость корпоративных центров обработки данных в решении задач бизнес-процессов. И, как следствие, важным становится вопрос защищенности и бесперебойной работы системы в целом.

На данный момент, качественно спроецированный ЦОД, с использованием актуального и мощного оборудования, способен решать ряд проблем, возникающих у компаний. Одними из важных вопросов в сферах связанных с потоком данных, является защита и отказоустойчивость при эксплуатации системы [2]. К решению этих задач необходимо относиться ответственно и находить решение, охватывающее всю покрывающую область, не нарушая целостность и работу системы.

Литература

1. Центры обработки данных (дата-центры) – Режим доступа: https://www.olly.ru/blog/centry-obrabotki-dannyh-data-centry – (Дата обращения: 15.04.2020).

2. Центры обработки данных: вчера, сегодня, завтра. – Режим доступа: https://compress.ru/article.aspx?id=18329 – (Дата обращения: 14.04.2020).

Киселева Ангелина Николаевна – студент 4-го курса кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: won.helium@gmail.com

Сафронов Виталий Владимирович (научный руководитель) – канд. техн. наук, доц., доцент кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: vitolik@bk.ru

УДК 531.1, 519.168

РЕШЕНИЕ ОБРАТНОЙ ЗАДАЧИ КИНЕМАТИКИ ЧЕТЫРЁХЗВЕННОГО МАНИПУЛЯТОРА МЕТОДОМ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

М. М. Коротков

Воронежский государственный университет

Введение

В материале этой статьи представлен один из способов реализации решения обратной кинематической задачи для четырёхзвенного манипулятора методом генетического алгоритма [1]. Под обратной задачей кинематики манипулятора понимается как поиск оптимального положения звеньев манипулятора в заданной точке, так и поиск оптимального пути и перемещения звеньев до требуемого положения. Очевидно, что эти две задачи связаны. В статье представлен обзор решения обеих задач. Генетический алгоритм позволит применять и синтезировать решения в зависимости от требования пользователя. Также рассмотрена оценка работоспособности и представлены предложения модификации.

1. О генетических алгоритмах

1.1. Описание метода

Генетический алгоритм работает аналогично работе эволюции живых организмов. Первым шагом генерируется некоторое количество особей. Как правило, особь – это строка каких либо характеристик или строка, в которой содержится информация о нужном нам решении. Это некоторая аналогия ДНК кода, в котором содержится информация о живом организме. Каждую особь оценивают специальной функцией приспособленности (выживаемости). Далее отбираются наилучшие особи, которые создадут потомков, которые заменят наихудшие. И так, шаг за шагом, каждое новое поколение, по заданной пользователем оценке оказывается лучше предыдущего [1].

Схема генетического алгоритма:

Рис. 1. Схема генетического алгоритма

Генетический алгоритм реализуется данной последовательностью действий:

1. Формирование начальной популяции.
2. Оценка особей популяции.
3. Отбор (селекция).
4. Скрещивание (кроссовер).
5. Мутация.

6. Формирование новой популяции.

7. Проверка на остановку. Если остановка не выполнена, то производится переход на шаг 2, иначе алгоритм заканчивает свою работу.

При каждой поставленной задаче, которую мы намерены решить методом генетического алгоритма, нам необходимо определить, что в данном случае является особью и по какому критерию мы будем её оценивать.

С помощью генетических алгоритмов чаще всего решают оптимизационную задачу. Пусть имеется некоторая функция $f(x_1, x_2, \ldots, x_n)$. Нам необходимо подобрать такую комбинацию $x_1, x_2, \ldots, x_n$, чтобы эта функция приняла максимальное значение. Функция $f(x_1, x_2, \ldots, x_n)$ называется функцией приспособленности или выживания (fitness function). $x_1, x_2, \ldots, x_n$ – это особь, оценку которой даёт фитнес-функция. Фитнес-функция должна принимать неотрицательные скалярные значения, по которому можно судить о качестве решения.

Если нам нужно найти минимум этой функции, то достаточно поменять условие отбора, т. е. или инвертировать данную функцию или отбирать нашу популяцию по минимальному значению.

Каждую популяцию необходимо кодировать. В классическом генетическом алгоритме, который впервые представлен в работе «Adaptation in Natural and Artificial Systems» (1975), была выбрана кодировка с помощью строки битов. Относительно требуемой точности результата была выбрана длина строки для хранения ряда чисел, которые представляют из себя решение. Таким образом, можно получить результат в дискретном виде и в заданном диапазоне. На практике особью часто являются набор чисел, строки и даже классы, и существует множество форм и вариаций самого алгоритма, например, «Эволюционный алгоритм».

Далее рассмотрим подробнее каждый шаг:

1) Формирование начальной популяции. На этом этапе формируется множество особей, из которых предстоит синтезировать необходимое нам решение, например, из множества неточных решений задач синтезировать более точное. На данном этапе необходимо определиться с количеством особей в поколении, это влияет и на скорость схождения задачи и оптимизацию. В классическом алгоритме начальная популяция генерировалась с помощью случайных чисел.

2) Оценка особей популяции. Далее с помощью заранее определённой функции приспособленности мы оцениваем все особи в этом поколении. Функция приспособленности определяет требуемое нами решение. Составление этой функции весьма творчески и в правильно реализованном алгоритме, при изменении функции приспособленности вы можете получить другое решение, необходимое для другой задачи.

3) Отбор (селекция). На данном этапе мы рассматриваем полученные оценки. Мы выбираем наихудшие варианты и отбрасываем их, чтобы заменить их новыми. Количество отбрасываемых элементов может быть разным, но в классическом алгоритме выбирался один наихудший. Также мы выбираем особей для следующего этапа, из которых мы будем получать новое с помощью функции скрещивания. Их необязательно 2 или несколько самых лучших, вполне может быть, что получать новых можно будет из лучшего и худшего или случайно выбранного элемента. Также часто применяется относительная оценка, когда элементы рассматриваются не по скалярным значениям оценочной функции, а относительно среднего значения популяции.

4) Скрещивание (кроссовер). Выбрав элементы на предыдущем этапе, нам необходимо получить новые. При желании даже из 2-х родителей можно получить неограниченное множество потомков, как и из n-го количество можно получить одного потомка. Функция скрещивания подбирается под определённое множество задач, так как, правильно составив функцию, можно добиться весьма быстрого схождения алгоритма. Полученные новые элементы записываются вместо отсеянных на предыдущем этапе.

5) Мутация. В классическом алгоритме мутация представляет из себя, то, что с определённой вероятностью (как правило, меньше одного процента) все особи, которые представляли из себя бинарное число, конвертировались. На практике мутировать могут не обязательно все особи. Мутация применяется для того, чтобы избежать нежелательных схождений алгоритма к не самому лучшему решению. Как правило, это или малое изменение некоторых особей, или с какой-то долей вероятности, или при определённых условиях.

6) Формирование новой популяции. После оценки, отбора, создания новых особей и, вероятно, мутаций, мы получаем новое поколение, которое оценим заново.

7) Проверка на остановку. После формирования нового поколения мы должны оценить, добились ли мы нужного нам решения. Это может быть оценка схождения алгоритма, когда мы видим, что разница значения максимума в оценочной функции в текущем поколении и прошедшем меньше заданного эпсилон или что цикл превысил максимальное количество итераций.

1.2. Оценка генетического алгоритма

Под оценкой генетического алгоритма имеется в виду его вычислительная сложность. Вычислительная сложность прямо зависит от количества итераций, проделанных для получения необходимого нам решения, это зависит от необходимой точности и скорости схождения. Генетический алгоритм сходится, когда в каждом следующем поколении максимально оценённая особь равномерно приближается к определённому экстремуму, достигнув которого, работу алгоритма следует остановить. Сразу необходимо оговориться, что чаще всего скорость схождения и достижения необходимого нам экстремума нелинейна, и анализ схождений и скорости достижения нужного нам результата – это большая область для изучения, так как данные процессы зависят от количества особей в начальной популяции, механизма отбора и скрещивания.

На каждом этапе эволюции должно быть вычислено значение оценочной функции каждой особи в популяции, после чего особи должны быть упорядочены по значениям их приспособленности, или должен быть проведен поиск необходимых значений. Величина вычислительной сложности зависит от количества особей в популяции. Также отдельно нужно вычислить мощность скрещивания и замены необходимых элементов. И ко всему необходимо добавить мутацию. Пусть N – количество особей в поколении, F – это вычислительная сложность оценочной функции, $S(N)$ – это вычислительная сложность отбора, C – вычислительная мощность скрещивания и M_{ut} – математическая сложность мутации одного элементе, N_m – количество особей, подверженных мутации, V_m – вероятность мутации, а N_g – количество итераций. Тогда вычислительная мощность генетического алгоритма P равна:

$$P \approx N_g \cdot \left(F \cdot N + S\left(N\right) + C\right) + M_{ut} \cdot N_m \cdot V_m. \quad (1)$$

2. Обзор решения задачи о положении звеньев манипулятора в заданной точке

2.1. Постановка задачи

Рассматривается четырехзвенный манипулятор на плоскости. Необходимо решить обратную задачу кинематики (ОЗК) с нахождением оптимального или близкого к оптимальному расположения звеньев манипулятора согласно выбранному критерию. Заметим, что задача динамики в данной работе не рассматривается, а масса звеньев будет выступать как заданный параметр для определения текущего расположения звеньев.

Пусть заданы координаты двух точек фиксации манипулятора $\left(x_0, y_0\right)$, $\left(x_4, y_4\right)$ (рис. 2) и известны длины четырех звеньев $i = 1 \ldots 4$. Необходимо найти координаты трех оставшихся вершин манипулятора $\left(x_1, y_1\right)$, $\left(x_2, y_2\right)$, $\left(x_3, y_3\right)$ и углов поворота звеньев $\alpha_1, \alpha_2, \alpha_3, \alpha_4$.

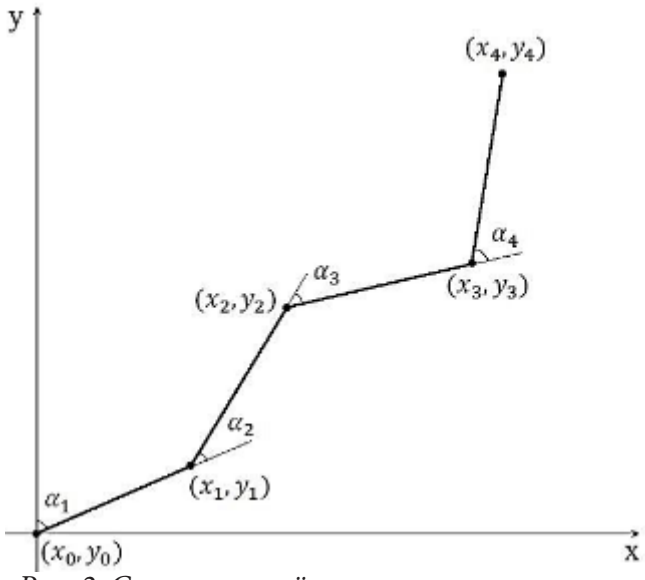


Рис. 2. Схема четырёхзвенного манипулятора

Заметим, что при всех известных углах поворота звеньев манипулятора, путём не хитрых математических вычислений, можно найти координаты (x_1, y_1), (x_2, y_2), (x_3, y_3). Тем самым, задача сводится к нахождению углов $\alpha_1, \alpha_2, \alpha_3, \alpha_4$.

2.2. Обзор решения задачи

Дадим определение особи в нашей задаче. Здесь возможны два варианта: набор координат (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4) или $\alpha_1, \alpha_2, \alpha_3, \alpha_4$, набор углов. В нашем случае это будет набор углов $\alpha_1, \alpha_2, \alpha_3, \alpha_4$. Так как при известных α_1 и α_4 мы можем однозначно определить α_2, α_3, то особью будет являться набор из двух углов α_1 и α_4. Этот набор мы обозначим с помощью двух векторов ξ_i, $u\psi_i$, где $i = 1..n$. ξ_i – это начальная популяция углов α_1, а ψ_i – начальная популяция углов α_4. n – это количество особей в данной популяции. Далее конкретизируем каждый из шагов алгоритма.

1. Формирование начальной популяции.

На данном этапе может быть использован любой из известных методов решения ОЗК, который позволяет получить заданное наперед количество решений. Все особи начальной популяции заполняются случайными значениями от нуля до π, с условием, что 2 соответствующих значения ξ_i и ψ_i, будут давать решение, удовлетворительное по отношению к необходимому условию корректного решения: расстояние между точками (x_1, y_1) и (x_3, y_3) должно быть меньше, чем $2 * l$, где l – длина стержня.

2. Оценка особей популяции.

Начальную популяцию необходимо оценить. Если у нас дан вектор начальной популяции $\xi_1, \psi_1, \xi, \psi_2, \ldots, \xi_i, \psi_i \ldots, \xi_n, \psi_n$, где $i = 1..n$, из него необходимо составить вектор оценочных значений o_i, где $i = 1..n$, который формируется исходя из заданных углов:

$$o_i = f(\xi_i, \psi_i), \qquad \text{где } i = 1 \ldots n, \tag{2}$$

$f(\xi_i, \psi_i)$ – является функцией выживаемости или оценочной функцией. Эта функция определяется пользователем или программой, исходя из условия задачи. Так же эта функция может зависеть от координат узлов манипулятора $f(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$. Эта функция должна возвращать скалярное значение, которое определяет качество решения. Чем больше число, тем более подходящее решение.

Вид этой функции определяется пользователем в соответствии с условием задачи. В нашей программе были представлены 4 функции. Если проводить селекцию, используя первую функцию, то результатом будут углы, при которых манипулятор примет вид «арки». Вторая, третья и четвёртая функция были взяты из книги.

3. Отбор (селекция).

Можно использовать любой из известных операторов селекции. В данной работе, был выбран пропорциональный отбор.

4. Скрещивание (кроссовер). Был предложен следующий вариант оператора скрещивания. Пусть выбраны два манипулятора-родителя. Тогда, чтобы получить первую группу потомков, необходимо у первого родителя задать некоторое допустимое смещение первого звена, а у второго родителя – четвертого звена. Тем самым это обеспечит 2 потомка следующему поколению.

5. Мутация. Мутация происходила в случае достижения определённой разницы между максимальным оценочным значением между предыдущим поколением и текущим. Тогда у половины популяции, не включая особь с максимальным значением оценочной функции, менялись углы на малое значение.

6. Формирование новой популяции.

В данной работе, был выбран вариант, в котором 2 наихудших значения заменялось потомками из 2-х лучших.

7. Проверка на остановку.

В данной задаче алгоритм останавливал работу после заданного числа итераций.

В итоге при использовании разных оценочных функций мы получали желаемый результат. На изображении можно видеть результат работы программы. Манипулятор в положении в форме «арки» находится в весьма удобном положении для захвата какого-либо элемента.

Рис. 3. Результат работы программы для формирования «арки»

3. Обзор решения задачи о передвижении манипулятора в заданную точку

3.1. Постановка задачи

Пусть заданы углы поворота звеньев $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ (рис. 4) и точка (x_t, y_t). Необходимо описать и найти оптимальные, последовательные и равноускоренные передвижения звеньев для установления конца манипулятора в точку (x_t, y_t).

Такая постановка даёт простор для фантазии. Во-первых, необходимо решить прямую задачу и определить текущее положение конца манипулятора. Во-вторых, необходимо описать передвижения манипулятора в заданную точку, принимая условие, что звенья двигаются поочерёдно, и главное – стоит вопрос об оптимальном решении. В-третьих, стоит вопрос о том, в какое положение необходимо поставить манипулятор в конечной точке.

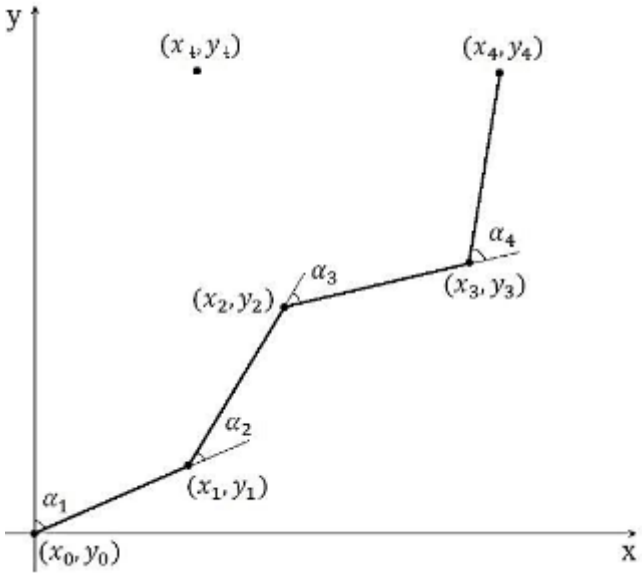


Рис. 4. Изображение к поставленной задаче

3.2. Предложенные методы решения задачи

Предлагается так же решить эту задачу методом генетического алгоритма. Это произойдёт в два этапа. В первом этапе, мы решим задачу и найдём оптимальное положение манипулятора в точке (x_t, y_t). Это решение даст нам набор из четырёх углов: $\beta_1, \beta_2, \beta_3, \beta_4$.

Вторым этапом, мы должны описать движение манипулятора из положения $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ в $\beta_1, \beta_2, \beta_3, \beta_4$ соответственно. Предлагается описать передвижение упорядоченным множеством из 2-х соответствующих элементов: угла и номера звена (k_i, δ_i), $i = 1..N$, где k_i – это номер поворачиваемого звена, δ_i – угол, на который должно повернуться текущее звено. Этот упорядоченный набор назовём особью.

Сформируем начальную популяцию. Формирование начальной популяции в нашем случае обусловлено. Для особи должно выполняется следующее условие:

$$\alpha_1 + sum\{\delta_i : k_i = j\} = \beta_j . \quad (3)$$

То есть каждое звено должно повернуться ровно до необходимого. Это будет главное условие в формировании начальной популяции.

Оценка решения. Естественно, критерии пройденного пути должны быть определены пользователем. Но нужно заметить, что оценка в задаче первого этапа и второго неразрывно связана. Если для нас важны минимальные перемещения, то в первом этапе это должно учитываться. Более интересный вопрос состоит во взаимном схождении первой и второй задачи.

Также стоит ряд вопросов и предложений о сходимости и оптимизации этих задач.

Заключение

В рассмотренной статье была предложена, поставлена и решена задача кинематики четырёхзвенного манипулятора с помощью генетических алгоритмов. Открыт ряд вопросов и предложений для улучшения реализации и дальнейшего изучения этой темы.

Литература

1. *Медведев, С. Н.* Разработка метода определения присоединенных параметров манипулятора с вращающимися звеньями на основе генетических алгоритмов / С. Н. Медведев, А. Ю. Яковлев, О. Г. Корольков // Вестник Воронежского государственного технического университета. – Т. 14, № 6. – 2018. – С. 8.

Коротков Михаил Михайлович – магистрант 1-го года обучения кафедры механики и компьютерного моделирования Воронежского государственного университета.
E-mail: mixan4ik@rambler.ru

Яковлев Александр Юрьевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры механики и компьютерного моделирования Воронежского государственного университета. E-mail: yakovlev@amm.vsu.ru

УДК 004.021

РАЗРАБОТКА АЛГОРИТМА ПОИСКА ЦЕНТРА ТЯЖЕСТИ НА ЦИФРОВОМ ИЗОБРАЖЕНИИ

В. А. Костин, С. Ю. Болотова

Воронежский государственный университет

Введение

В изобразительном искусстве и в фотографии существует понятие центра тяжести. Это точка, где сосредоточен вес предметов, по отношению к которой остальные предметы располагаются таким образом, чтобы сохранить равновесие. Для выполнения своих работ художники вручную находят композиционный центр изображения и выставляют сцену таким образом, чтобы центр находился в нужном месте.

В данной работе рассматривается алгоритм поиска центра тяжести на цифровом изображении, модификация для работы в реальном времени и его программная реализация.

Алгоритм поиска центра тяжести

Рассмотрим основные алгоритмы, которые используются для достижения поставленной задачи. Изображение может иметь большое разрешение, и обработка может занимать большое количество времени, и чтобы применить алгоритм, его необходимо обработать.

На первом шаге выполняется предварительная обработка изображения – выделение граней и удаление шумов. В первую очередь изменяется цветовой режим, оно преобразуется в оттенки серого, т.к. цветовые характеристики не используются в алгоритме и не требуются для дальнейших преобразований.

Далее необходимо выделить области с сильным контрастом интенсивности, т. е. грани. Для выделения граней используется алгоритм Джона Ф. Кенни [2]. Несмотря на то, что алгоритм выделения границ удаляет шумы, могут остаться границы небольшого размера, которые так же можно принять за шум, так как они являются незначительными и могут помешать верному определению контуров. Для данной цели используется билатеральный фильтр и растягивание контуров [2].

Определение. Билатеральный фильтр – нелинейный оператор, сохраняющий края и смягчающий шум [2]. Он заменяет интенсивность каждого пикселя средневзвешенным значением интенсивности из соседних пикселей.

Определение. Расширение (растягивание) контуров – операция, расширяющая изображение с использованием матричного фильтра. Применение фильтра к изображению заключается в поэлементном умножении.

После выделения граней, необходимо определить, какие из них представляют собой грани объектов переднего плана, а какие – фоновых объектов. Выполнение такой сегментации изображения на основе различных интенсивностей в передних и задних областях изображения обеспечивает пороговое преобразование [4]. Не все изображения могут быть аккуратно сегментированы на передний план и фон с использованием простого порогового значения, поэтому простой случай порогового преобразования не всегда подходит, и применяется динамическое преобразование с локальным пороговым значением.

Для того, чтобы ускорить поиск контуров на изображении, выполняется заливка замкнутых крупных участков с помощью маски. Пиксели, которые не подвержены действию заливки, обязательно находятся внутри границы. Такое изображение при инвертировании и сочетании с пороговым изображением дает передний план изображения. Пример результата работы описанных выше алгоритмов изображен на рис. 1.

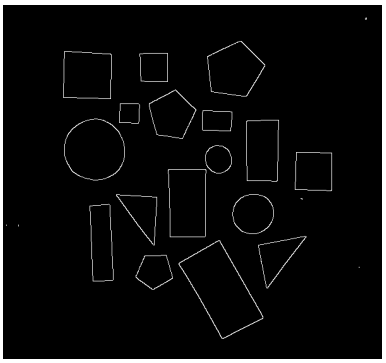


Рис. 1. Результат обработки изображения

После обработки выполняется поиск контуров. Для этого используется алгоритм следования границам [4], описанный Сатоши Сузуки и Кейчи Абе. Данный алгоритм работает с изображением как с набором компонент (наборов пикселей) и границ, имеющих иерархическую структуру. Алгоритм обходит границы на изображении, помечая пиксели номерами в зависимости от того, находятся ли они между 0-компонентой и 1-компонентой. На выход алгоритм отдает контуры, найденные на изображении, а также их иерархию.

Далее производится поиск центров найденных контуров. Координаты центра контура ищутся на основе его моментов [2].

Определение. Момент – это суммарная характеристика контура, рассчитанная суммированием всех пикселей контура. Момент $(p,\ q)$ определяется следующим образом:

$$m_{(p,q)} = \sum_1^n I(x,y)x^p y^q, \tag{1}$$

где p – порядок x, q – порядок y, где порядок означает мощность, на которой соответствующий компонент взят в сумме с другими компонентами.

Для решения поставленной задачи необходимо найти центральные моменты, которые определяются по формулам:

$$X_c = \frac{m_{10}}{m_{00}},\quad Y_c = \frac{m_{01}}{m_{00}}. \tag{2}$$

После того, как найдены центральные моменты контуров объектов на изображении, необходимо найти основной контур на изображении. Так как точек может быть много, среди центров тяжести нужно найти крайние точки. Наименьшее множество таких точек называется выпуклой оболочкой [1]. Пример выпуклой оболочки представлен на рис. 2.

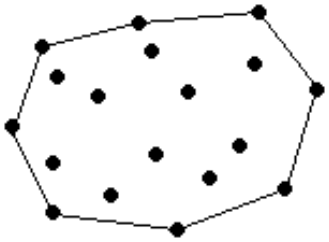


Рис. 2. Пример выпуклой оболочки контура

Для поиска выпуклой оболочки используется алгоритм Склански. Он проявляет себя как самый быстрый на случайном наборе точек, а наихудшее поведение демонстрирует в случае, если заданные точки уже образуют выпуклый многоугольник.

На основе найденной выпуклой оболочки формируется контур, для которого применяется ранее описанный алгоритм поиска центра, и, т.о. Будет найден центр тяжести изображения. Пример результата работы алгоритма на статическом изображении представлен на рис. 3.

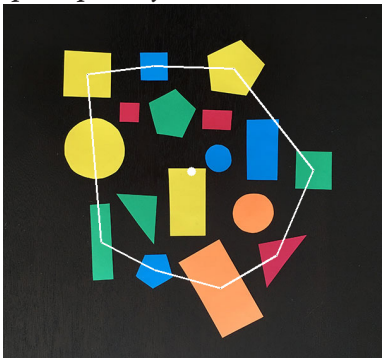


Рис. 3. Результат работы алгоритма – основной контур и центр тяжести

Модификация для работы в реальном времени

Для работы в реальном времени алгоритм подвергся модификации. Выполняется проверка, произошло ли движение объектов в видоискателе или сам видоискатель. Для проверки факта движения используется критерий совпадения дескрипторов. Проверка осуществляется на основе доли совпадения дескрипторов текущего изображения I, полученного с камеры, и изображения I_{prev}, на котором был найден контур в предыдущий раз. В том случае, если доля совпадения дескрипторов этих изображений будет меньше 60 %, то будем считать, что камера сдвинулась и положение объектов на изображении поменялось. Пример такого случая изображен на рис. 4 и рис. 5.

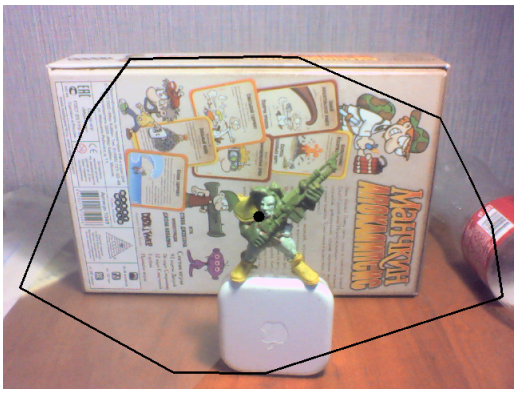


Рис 4. Контур и центр до движения камеры

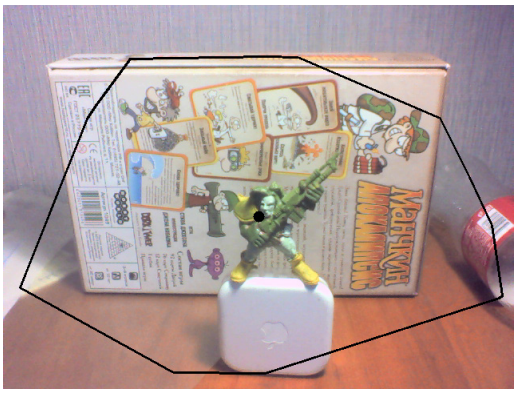


Рис 5. Контур и центр после движения камеры

Заключение

В результате проведенной работы было реализовано приложение со следующими возможностями: поиск центра тяжести на статическом изображении, поиск центра тяжести на изображении, полученном с камеры устройства в реальном времени наглядное визуальное отображение результата работы программы.

Данный алгоритм может быть применен в сферах изобразительного искусства и фотографии для поиска композиционного центра сцены, в системах компьютерного зрения для отслеживания объектов с целью фиксации видоискателя на основном объекте сцены, для отслеживания в реальном времени объектов с целью сохранения их в устойчивом состоянии.

Литература

1. Препарата, Ф. Вычислительная геометрия: Введение / Ф. Препарата, М. Шеймос. Пер. с англ. – М. : Мир, 1989. – 478 с.

2. Красильников, Н. Цифровая обработка 2D- и 3D-изображений / Н. Красильников. – СПб. : БХВ-Петербург, 2011. – 863 с.

3. Suzuki, Satoshi. Topological Structural Analysis of Digitized Binary Images by Border Following. / Satoshi Suzuki, Keiichi Abe // Computer vision, graphics, and image processing. – 1985. – Vol. 30 – P. 32–46.

4. Shapiro, Linda G. Computer Vision / Linda G. Shapiro, George C. Stockman. – Pearson, 2002. – 608 p.

Костин Владимир Алексеевич – магистрант 2-го года обучения кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: ollkostin@gmail.com

Болотова Светлана Юрьевна (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: bolotova.svetlana@gmail.com

УДК 681.3.06

ИНТЕРНЕТ-ТЕХНОЛОГИИ В АКТИВИЗАЦИИ РАБОТЫ СТУДЕНЧЕСКОГО СОВЕТА ФАКУЛЬТЕТА ПРИКЛАДНОЙ МАТЕМАТИКИ, ИНФОРМАТИКИ И МЕХАНИКИ

И. Д. Коток

Воронежский государственный университет

В 2019 году факультет прикладной математики, информатики и механики отпраздновал полувековой юбилей. Как отметил декан, профессор Шашкин А. И.: «Факультет ПММ уже который год занимает лидирующие позиции по всем направлениям. Студенты факультета лучшие во всем – в учебе, науке, спорте, творческой самодеятельности» [5].

В 2018 года на факультете прикладной математики, информатики и механики был реорганизован студенческий совет, в связи с растущим интересом к общественной деятельности студентов нашего факультета. На первом собрании было решено вести работу в нескольких направлениях:

- спортивная деятельность,
- пресс-центр,
- научная деятельность,
- работа с абитуриентами.

Спортивная деятельность – направление, которое курирует физорг факультета. Основная задача данного направления помощь в организации и проведении спортивных мероприятий кафедре физического воспитания и спорта и спортивному клубу ВГУ. Информирование студентов и аспирантов о тренировках и соревнованиях, в которых они могут принять участие или быть задействованы в качестве волонтеров, судей и болельщиков. Руководителем данного направления является физорг факультета.

Пресс-центр – направление, которое организует информационную поддержку факультета: сайт, социальные сети. Помимо этого, данное направление занимается освещением мероприятий, связанных с факультетом и вузом, ответственно за работу с пресс-центром ВГУ, с октября 2018 года занимается записью подкастов.

Научная деятельность – направление, которое занимается оказанием помощи отстающим студентам, проведением консультаций, оказанием помощи в организации и проведении научных международных конференций, научных фестивалей и студенческих олимпиад.

Работа с абитуриентами – это одно из самых важных направлений факультета. Его деятельность в основном связана со школьниками, которые планируют сделать выбор их дальнейшей специальности. Оно занимается помощью деканату в организации и проведении дней открытых дверей, а также информационных компаний, направленных на выпускников школ.

Студенческий совет ПММ активно сотрудничает с администрацией ВГУ, деканатом, компаниями партнеров. Каждый член этой организации старается внести свой вклад в улучшение работы факультета, сделать работу деканата легче, заинтересовать студентов развиваться в различных сферах, начиная наукой, заканчивая спортом. Механизмы, которые применяются, благоприятно влияют на межличностные отношения, как среди студентов, так и помогают наладить диалог преподаватель – студент.

1. Использование современных интернет технологий в работе факультета

Со времени создания студенческого совета факультета его руководящий состав решил придерживаться современных тенденций, которые могут облегчить работу, как им самим, так и сотрудникам факультета. Информационный поток становится более разнообразным. Многие современные тенденции были уже апробированы, поставлены на поток и работают уже достаточно продолжительное время.

1.1. Интеллектуальные викторины

В ноябре 2018 года в Воронежском государственном университете на базе факультета прикладной математики, информатики и механики был проведен первый кубок факультета ПММ по интеллектуальным играм, приуроченный к пятидесятилетию факультета и столетию ВГУ. Было решено возродить это интересное мероприятие, которое проводилось ранее, на двадцатипятилетний юбилей факультета.

Данный турнир проходил по правилам спортивной версии интеллектуальной игры «Что? Где? Когда?». Данная игра приобрела огромную популярность в нашей стране с самого первого показа ее на телеэкранах в 1975 году. Она очень популярна среди людей всех возрастов и поколений. Игры подобного типа развивают креативное мышление человека, что очень важно для дальнейшего жизненного пути, также способствуют развитию коммуникативных навыков, а самое главное для любой игры, помогает в окружении друзей и близких отлично провести свой досуг.

Желание поучаствовать в соревнованиях изъявило более 16 команд с разных факультетов Воронежского государственного факультета. К сожалению, на тот момент студенческий совет мог подтвердить участие лишь 8 командам. Из-за отсутствия альтернативных решений текущей проблемы, было решено обратиться к приложению по генерации случайных чисел, которое помогло отобрать 8 команд счастливчиков, получивших возможность принять участие в турнире [3–5].

Данная проблема подтолкнула оргкомитет в дальнейшем найти более справедливый способ отбора команд. Было решено добавить заочный этап, который включал в себя онлайн отбор участников на очный этап. В результате, создано новое направление на базе студенческого совета, которое занимается проведением интеллектуальных викторин. На данный момент было опробовано более семи различных соревнований подобного типа, которые были хорошо встречены, как со стороны студентов, так и со стороны преподавателей нашего факультета.

Переработав большое количество материала было решено, что необходим сайт для проведения, как заочного этапа, так и хранения информации об очных турнирах: фотоотчеты, списки вопросов, задействованные на проводимых турнирах, и участников команд.

На данный момент мы имеем достаточно много различных интернет ресурсов, которые занимаются сбором вопросов по различным интеллектуальным викторинам. К сожалению, платформы, которая могла воплотить все идеи и задумки оргкомитета, нет. Задача является узкоспециализированной и требует индивидуального подхода.

Веб-приложение разрабатывается в среде Visual Studio 2019 со встроенной платформой .Net Core. В качестве языка программирования был выбран язык C#. Данный выбор обусловлен современностью этого языка программирования, количеством документации по текущей технологии и наличием в среде разработки широкого набора средств для реализации данного приложения [1, 2].

1.2. Работа пресс-центра студенческого совета

В октябре 2018 года при поддержке деканата нашего факультета пресс-центром студенческого совета ПММ был записан первый подкаст со студентом-активистом факультета. На данный момент отснято уже 15 сюжетов и останавливаться никто не собирается. Главными героями очередного выпуска успели побывать ректор Воронежского государственного университета Дмитрий Александрович Ендовицкий, декан нашего факультета Александр Иванович Шашкин, несколько преподавателей, выпускников и сотрудников деканата, партнеры факультета, сотрудники университета и, конечно же, простые студенты ПММ.

Помимо этого, активистами студенческого совета были разработаны различные чат-боты, которые были установлены в официальном сообществе студенческого совета в социальной сети «ВКонтакте».

Данные боты сейчас помогают решать следующие задачи:

• по фамилии преподавателя выводит его ФИО, также указывает кафедру, на которой работает данный сотрудник, и номер ее аудитории;

• вывод информации, которая на данный момент неделя числитель или знаменатель.

Сейчас планируется разработка новых чат-ботов, один из них должен выводить текущее расписание данной группы или преподавателя. Это поможет облегчить жизнь многим студентам и преподавателям, чтобы не тратить время на поиски этой информации на сайте факультета.

1.3. Спортивная деятельность

В направление по спортивной деятельности, к сожалению, пока нововведений нет. Сейчас решается вопрос о создание веб-приложения, которое будет содержать всю информацию о спортивной жизни ВГУ:

• отдельное место под все факультеты ВГУ, для размещения локальной информации о каждом из факультетов, ведение статистики спортсменов, расписание секций, фото-видео отчеты спортивных мероприятий;

• расписание занятий по физической культуре и спорту;

• методические материалы и обучающие видео материалы;

• освещение деятельности спортивного клуба ВГУ.

Таким образом, работа по развитию студенческого совета факультета идет полным ходом. Его руководство старается по максиму внедрять в свою деятельность современные интернет технологии, чтобы сделать все ресурсы, которые они имеют в своем распоряжении, максимально доступными для студентов, преподавателей, сотрудников, партнеров и абитуриентов.

Сейчас активно разрабатывается сайт для проведения интеллектуальных викторин и турниров этой направленности. Это стало особенно актуально в связи с тяжелой эпидемиологической обстановкой в стране и мире. Идет работа по поиску альтернатив для записи новых подкастов, чтобы соблюдать действующие на данные момент самоизоляционные меры и поддерживать качество аудио записи того или иного интервью на высоком уровне.

Литература

1. *Албахари Джозеф, Албахари Бен.* C# 7.0 Справочник. Полное описание языка.: Пер. с анг. – СПб. : ООО «Диалектика», 2019. – 1024 с.

2. .NET Documentation | Microsoft Docs. – Режим доступа: https://docs.microsoft.com/en-us/dotnet/

3. *Коток, И. Д.* Разработка программного обеспечения для проведения интеллектуальных компьютерных игр студенческим советом факультета прикладной математики, информатики и механики / Актуальные проблемы прикладной математики, информатики и механики: сб. тр. Междунар. науч.-техн. конф. (Воронеж, 11–13 ноября 2019 г.). – Воронеж : Издательство «Научно-исследовательские публикации», 2020. – С. 443–444.

4. *Коток И. Д.* Разработка программного обеспечения для проведения интеллектуальных викторин / Информационные технологии в образовательном процессе вуза и школы: материалы XIV Всероссийской научно-практической конференции (Воронеж, 25 марта 2020 г.). – Воронеж: Воронежский государственный педагогический университет, 2020. – С. 171–175.

5. 50 лет факультету прикладной математики, информатики и механики / Редакционная коллегия: А. И. Шашкин, О. Ф. Ускова, Т. Г. Богомолова. – Воронеж : ООО ИПЦ «Научная книга», 2019. – 26 с.

Коток Игорь Дмитриевич – студент 4 курса 61 группы кафедры математического обеспечения ЭВМ факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: tigariskot@gmail.com

Ускова Ольга Фёдоровна (научный руководитель) – канд. техн. наук, профессор кафедры математического обеспечения ЭВМ факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: nat-uskova@mail.ru

УДК 004.89

ВЫБОР МОДЕЛИ МАШИННОГО ОБУЧЕНИЯ ДЛЯ АНАЛИЗА ТЕКСТОВЫХ ДАННЫХ

Ф. Ю. Лебедев

Воронежский государственный университет

Введение

Часто родителям, учителям, работодателям и пр. необходимо осуществлять контроль деятельности в интернете. Поэтому разработка анализатора, который будет запрещать некоторые запросы в поисковике интернет браузера, является важной задачей, решением которой с радостью воспользуются многие люди.

Необходимо анализировать интернет запросы для того, чтобы определить, является ли запрос приемлемым или нет. Под приемлемостью будем понимать, удовлетворяет ли запрос критериям допустимости для определённой группы пользователей. Например: тип группы – возрастная, возраст – дети 0–7 лет, критерий допустимости – социальные сети. Так запрос «Смотреть мультфильм Маша и Медведь 5 серия» будет приемлемым, а запрос «Форсаж 9 фото Instagram» будет неприемлемым. Более того, данный анализ в итоге должен проходить автоматически и без участия эксперта, который изначально будет поддерживать работу анализатора.

Для решения задачи проанализируем модели машинного обучения, которые в большей степени могут подойти для реализации поставленных требований, и выберем одну из них для дальнейшего использования.

1. Виды машинного обучения

Предполагается, что работа анализатора будет поддерживаться экспертом, следовательно, остановимся на методах машинного обучения с учителем. Обучение без учителя не подойдёт нам по определению, а другие виды машинного обучения не подходят из-за своей узконаправленности. Например, для обучения с подкреплением важнейшую роль играет реакция окружающей среды, а для части других задач существенным фактором является время.

2. Обучение с учителем, группы задач

Рассмотрим задачи, для решения которых применимо обучение с учителем. Тип задач – обучение с учителем подразумевает, что у нас будет пара значений «объект, ответ». Исходя из условий нашей задачи, объектом будет строка – запрос пользователя, а ответом – приемлемость, т. е. «да» или «нет». Рассмотрим основные подтипы задач обучения с учителем и выберем наиболее соответствующий нашей задаче. Результаты анализа подтипов представлены в табл. 1.

Таблица 1

Задачи обучения с учителем

Тип задач	Особенность
Задача классификации	Множество ответов конечно
Задача регрессии	Ответом может быть действительное число или числовой вектор
Задача ранжирования	Ответы получаем и рассматриваем на целом множестве объектов, затем сортируем их по значениям
Задача прогнозирования	Объектами являются отрезки времени. Обрабатываем эти объекты и делаем прогноз на будущее

Решаемая задача требует разбить запросы пользователя на два множества, «удовлетворяет» и «не удовлетворяет», значит, работа будет вестись с бинарным множеством ответов, причём объектами являются строки. Следовательно, самый подходящий нам тип задач – классификация, точнее бинарная классификация.

3. Выбор модели

3.1. Общие сведения о моделях

Из-за большого разнообразия решаемых задач, разнообразие моделей машинного обучения крайне велико. В основе моделей лежат принципы решения задач из различных наук (математика, информатика, психология, филология, биология и др.), которые в корне отличаются, поэтому сами модели тоже отличаются очень сильно. Но всё-таки можно выделить несколько главных типов моделей:

1) геометрические модели;
2) вероятностные модели;
3) логические модели;
4) текстовые модели.

Важно понимать, что эти типы не являются взаимоисключающими и конкретная модель может одновременно принадлежать сразу нескольким типам.

3.2. Анализ моделей

3.2.1. Геометрические модели

Линейный классификатор, изображённый на рис. 1. – это геометрический классификатор.

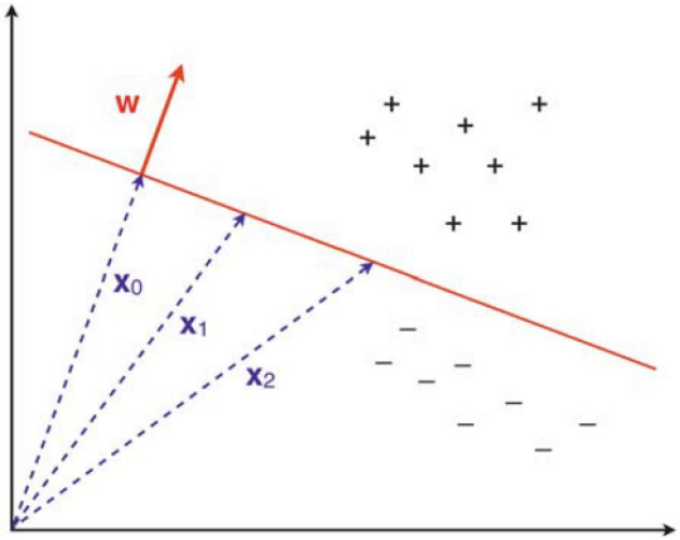


Рис. 1. Пример линейной классификации в двух измерениях. Прямая линия отделяет положительные результаты от отрицательных. Она определяется уравнением $w * x_i = t$, *где w – вектор, перпендикулярный решающей границе и направленный в сторону положительных результатов,* t *– порог принятия решений, а* x_i *– вектор, оканчивающийся на решающей границе*

В нашей задаче множество входных данных практически безгранично, если рассматривать входной запрос как множество зачастую несвязанных слов. Главное достоинство геометрических классификаторов – это их наглядность. Но при огромном количестве условий, т. е. входных данных, размерность геометрической модели доходит до сотен и даже тысяч пространств,

изображение чего практически невозможно. Линейная разделимость для задачи бинарной классификация текста применима, но чрезмерное множество входных данных не позволит нам чётко изобразить классификацию в n-мерном пространстве.

В задаче анализа текстовых запросов, вариативность которых практически безгранична, получаем n-мерную геометрическую модель. Так как n – чрезмерно большое значение, то теряются главные достоинства геометрической модели: простота и эффективность, следовательно, геометрическая модель нам не подходит.

3.2.2. Вероятностные модели

Поскольку в задаче, обрабатывающей пользовательские запросы, эти запросы заранее неизвестны и в них могут содержаться слова, которые впервые встретились, неизбежна неопределённость. А обработка неопределённости – это одна из главных задач вероятностных моделей.

Так бинарную классификацию можно представить в виде принадлежности к единственному классу, относятся ли исследуемые данные к единственному классу или нет. Формулу (1) можно толковать как бинарную классификацию, вероятность принадлежности данных «приемлемому» запросу.

$$P(A) = \frac{m}{n}. \tag{1}$$

В простейшем случае вероятность события A («приемлемость» запроса) равна частному количества «приемлемых» слов в запросе к общему числу слов n. «Приемлемость» отдельных слов изначально будет оцениваться экспертом.

Исходя из формулы (1), если событие A зависит от двух событий B и C, т. е. исследуемый запрос разбит на два подзапроса, получим формулы (2) и (3):

$$P(A) = P(B + C) = P(B) + P(C) = \frac{m_1}{n} + \frac{m_2}{n} = \frac{m}{n} \text{ для } m = m_1 + m_2 \tag{2}$$

$$P(A) = P(B + C - BC) = P(B) + P(C) - P(BC) = \frac{m_1}{n} + \frac{m_2}{n} - \frac{m_1 m_2}{n} = \frac{m}{n} \tag{3}$$

$$\text{для } m = m_1 + m_2 - m_1 m_2.$$

Формула (2) – это частный случай, когда события B и C независимы, а формула (3) покрывает ситуацию, когда эти события могут быть зависимы. Так как разделение запроса на зависимые события (одно или несколько слов попадают в разные группы) нецелесообразно, дальнейшему анализу будет подвержена формула (2).

Событие A можно разбить максимальным образом на k событий, где k – количество слов в запросе. Посчитать вероятность «приемлемости» отдельных слов по средствам математических алгоритмов невозможно, поэтому нам потребуются эксперт и текстовая модель.

3.2.3. Логические модели

Логические модели можно выразить на языке правил.

Такие правила часто реализуются в виде дерева, называемого деревом признаков или деревом решений (рис. 2).

Задача логической модели – итеративно разбить пространство объектов. Признаки, например, «В запросе есть неприемлемые слова», используются для итеративного разбиения пространства объектов. Листы дерева – это области пространства объектов, в нашем примере таких областей всего две: «Разрешить» и «Запретить», пример соответствует бинарной классификации. Древовидную модель, представленную на рис. 2 на языке программирования Java можно записать в следующем виде (Листинг 1).

Листинг 1

```
public boolean acceptable (String[]inputQuery, Set < String > unacceptableWords) {
      if (inputQuary.length() > 1) {
        if (contains(inputQuery, unacceptableWords)) {
          if (unacceptableCtn(inputQuery, unacceptableWords) <
              acceptableCtn(inputQuery, unacceptableWords)) {
            return true;
          } else {return false;}
         else {return true;}
        }
      } else if (inputQuary.length() == 0) {
        return true;
      } else if (isAcceptable(inputQuary[0]) {
        return true;
      } else {return false;}
    }
```

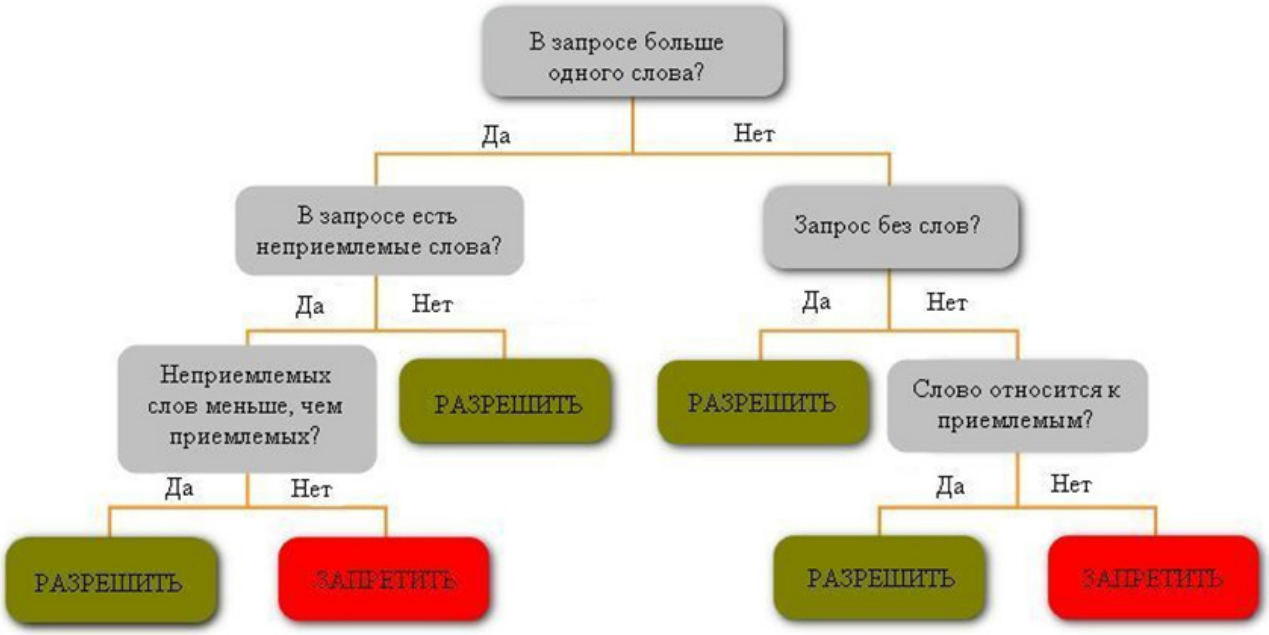


Рис. 2. Пример логической модели, которая определяет, «разрешить» выполнение запроса или «запретить»

Главным недостатком логических моделей является их небольшая гибкость, каждый признак разделяет объекты на две части. В отличие от вероятностных моделей, где можно настроить проходное значение, например «Разрешить» при $P(A) = 0.6$, логические модели всегда имеют дело со значением 0.5. Сложность и запутанность программирования логических моделей является ещё одной проблемой. Каждое новое условие добавляет дополнительную проверку, в представленном выше коде всего пять проверок, и уже наблюдается его нечитабельность; при десяти проверках разобраться в модели будет уже проблематично.

3.2.4. Текстовые модели

Практически все текстовые модели создаются под определённый естественный язык. Наша модель будет обрабатывать русский или английский. В машинном обучении есть раздел – обработка естественного языка (ОЕЯ), программные реализации ОЕЯ нам и потребуются.

В нашей задаче не планируется использовать полноценный анализ входного запроса, включая связи слов, пунктуацию, междометия и т. д., так как обработка запроса должна проходить в режиме реального времени и минимально нагружать систему. Более того, хранилища результатов полноценного анализа запросов будет на порядок объёмнее хранилища анализа частичных данных.

Как мы уже обсуждали, нам понадобится анализировать лишь три части речи, дополняя этот анализ синонимами. Данный подход лишь немного повлияет на продуктивность, так как для определения сути запроса не нужно быть лингвистом. Так же рациональным решением будет внедрение синонимов в проектируемую модель, так как они являются мощным инструментом для решения неопределённости. На основе синонимов, «приемлемость» которых уже установлена, можно установить «приемлемость» нового слова, не прибегая к помощи эксперта. Более того, синонимы расширят наше хранилище, что позволит проводить более точный анализ, так мы можем учесть часть фразеологизмов, например «презренный металл».

Кроме того при проектировании текстовой модели, правильным подходом будет использование лемматизации. Работа с базой данных, состоящей из сотен тысяч словоформ куда сложнее, чем с несколькими тысячами слов, приведённых к словарной форме.

Заключение

Итак, перед нами стоит задача бинарной классификации текстовой информации, для которой на начальных этапах необходима помощь эксперта, затем решение будет функционировать автономно. Для выбора нужной модели обратимся к табл. 2, в которой приведены особенности существующих моделей.

Нашим решением будет скомпонованная модель, так как особенностей любой одной модели недостаточно. Учитывая, что мы работаем с текстом, и он может быть крайне вариативным, а так же масштабируемость и гибкость вероятностной модели, самое верное решение – модель, включающая в себя принципы текстовой модели и логику вероятностной модели.

Таблица 2

Типы моделей

Тип модели	Особенность
Геометрическая модель	Наглядность при небольшом числе условий, большая вариативность, сложность при большом количестве условий
Вероятностная модель	Масштабируемость и простота при большом количестве условий
Логическая модель	Максимальная простота, сложность при проектировании широкого функционала
Текстовая модель	Незаменима при работе с текстом

Литература

1. *Петер Флах*, Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных / пер. с англ. А. А. Слинкина. – V. : LVR Пресс, 2015. – 400 с.

2. *Stuart Russel, Peter Norvig.* Artificial Intelligence. A modern Approach. – Third Edition, 2009. – 1152 с.

3. *Григорий Алексеев,* Введение в машинное обучение [Электронный ресурс] – Режим доступа: https://habr.com/ru/post/448892/ (Дата обращения: 20.03.2020).

4. Создатели сайта machinelearning.ru при использовании литературы, указанной в электронном ресурсе. Машинное обучение [Электронный ресурс] – Режим доступа: http://www.machinelearning.ru/wiki/index.php?title=Машинное_обучение (Дата обращения: 20.03.2020).

5. *Tyler Watkins*. Probability for Machine Learning Discover How To Harness Uncertainty With Python [Электронный ресурс] – Режим доступа: https://machinelearningmastery.com/probability-for-machine-learning/ (Дата обращения: 05.04.2020).

6. *Ventsislav Yordanov*. Основы Natural Language Processing для текста / пер. с англ. Nvpushkarskiy2 [Электронный ресурс] – Режим доступа: https://habr.com/ru/company/Voximplant/blog/446738/ (Дата обращения: 11.04.2020).

Лебедев Федор Юрьевич – магистрант 2-го года обучения кафедры математического обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: www.realmanfyodor@mail.ru

Воронина Ирина Евгеньевна (научный руководитель) – д-р техн. наук, проф., профессор кафедры программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: irina.voronina@gmail.com

УДК 004.021

РАЗРАБОТКА УНИВЕРСАЛЬНОГО МОДУЛЯ КОНФИГУРАЦИИ 1С

Д. С. Литвинов

Воронежский государственный университет

Введение

«1С: Предприятие» – среда программ фирмы «1С», предназначенная для автоматизации деятельности предприятия. «1С: Предприятие» состоит из двух основных взаимосвязанных частей: прикладное решение (конфигурация) и технологическая платформа. Конфигураций на данный момент очень много (1С: Бухгалтерия, 1С: Управление торговлей, 1С: Зарплата и управление персоналом и т.д.), а технологическая платформа одна. В свою очередь, конфигурация устанавливается поверх платформы, после чего пользователи уже могут приступать к работе.

Гибкость платформы позволяет применять «1С: Предприятие» в разных областях: автоматизации производственных и торговых предприятий, бюджетных и финансовых организаций и т. д.; поддержки оперативного управления предприятием; автоматизации организационной и хозяйственной деятельности; расчета заработной платы и управления персоналом и других областях [1].

1. Универсальный модуль интеграции с Active Directory

Хоть и основным предназначением «1С: Предприятия» является ведение учета на предприятии, на данный момент возможности технологической платформы ушли намного дальше, чем просто обработка и проведение документов, формирование отчетов и т. д. В последних версиях платформы реализованы достаточно мощные возможности взаимодействия с web-технологиями, возможна разработка мобильных приложений. Благодаря такому стремительному развитию технологической платформы, стали возможны обмены и интеграции не только между несколькими конфигурациями, но и с внешними объектами. Благодаря этим возможностям, «1С: Предприятие» становится уже не только программой для ведения учета, но и может стать целой мастер-системой по учету пользователей сервера, что делает модуль конфигурации универсальным.

Контролем и ограничением возможностей пользователей, рабочих мест пользователей, различных сетевых устройств и серверов сети управляет Active Directory (AD). ***AD*** – это иерархически организованное хранилище данных об объектах сети, обеспечивающее удобные средства для поиска и использования этих данных. На рисунке 1 изображена общая схема AD. Компьютер, на котором работает AD, называется контроллером домена. С AD связаны практически все административные задачи. AD объединяет физическую и логическую структуру для компонентов сети. Логические структуры AD помогают организовывать объекты каталога и управлять сетевыми учетными записями и общими ресурсами. К логической структуре относятся следующие элементы:

- организационное подразделение (organizational unit) – подгруппа компьютеров, как правило, отражающая структуру компании;
- домен (domain) – группа компьютеров, совместно использующих общую БД каталога;

- дерево доменов (domain tree) – один или несколько доменов, совместно использующих непрерывное пространство имен;

- лес доменов (domain forest) – одно или несколько деревьев, совместно использующих информацию каталога.

Рис. 1. Общая схема AD

Физические элементы помогают планировать реальную структуру сети. На основании физических структур формируются сетевые связи и физические границы сетевых ресурсов. К физической структуре относятся следующие элементы:

- подсеть (subnet) – сетевая группа с заданной областью IP- адресов и сетевой маской;

- сайт (site) – одна или несколько подсетей. Сайт используется для настройки доступа к каталогу и для репликации.

Немаловажным плюсом Active Directory является отказоустойчивость и распределенная база данных. Служба каталогов может охватывать один или несколько доменов, т. е. областей безопасности сети. Если несколько доменов связаны доверительными отношениями, имеют общую схему, одну и ту же конфигурацию и глобальный каталог, это значит, что домены объединены в дерево и образуют общее пространство имен. Несколько деревьев могут быть объединены в лес – набор деревьев, которые не образуют общего пространства имен, но связаны доверительными отношениями. Говоря о возможностях Active Directory, следует отметить такие, как делегирование (предоставление администратором прав администрирования отдельным пользователям и группам на контейнеры и поддеревья) и наследование (права доступа к контейнеру распространяются на все его дочерние контейнеры) [2].

На стороне 1С можно разработать модуль интеграции с AD, который будет исполнять практические любые запросы заказчика. Возможно создать в модуле справочник, который будет содержать все учетные записи пользователей AD или же все рабочие места. Возможно создать документ, который при приеме нового сотрудника в штат будет автоматически создавать нового пользователя AD. При этом возможности создания, удаления или перемещения сотрудника непосредственно из AD так же остаются, а контроль за синхронизацией данных 1С из AD будет установлен в 1С за счет выполнения регламентного задания, которое будет выполняться с определенным интервалом времени.

Приведем более конкретный пример. Некоторая организация имеет в своем штате более 1000 сотрудников. В определенный момент возникает задача: проводить внутренние рассылки на корпоративные почтовые ящики, которые выдаются каждому сотруднику при приеме на работу, и проверить, заблокированы ли учетные записи уволенных сотрудников. Но при этом в 1С нет информации по почтовым адресам, но только там возможно понять, работает ли ещё сотрудник или нет. Эту задачу можно достаточно просто решить созданием обработ-

ки внутри программы 1С, которая будет синхронизировать пользователей AD, записывать в карточку сотрудника в справочнике «Сотрудники» корпоративный e-mail сотрудника, затем сопоставленные данные будут записаны в регистре сведений. На рисунке 2 приведен пример возможного внешнего вида обработки для заполнения сопоставления учетных записей AD и сотрудников. И когда нам потребуется провести проверку и блокировку недействующих учетных данных AD, достаточно будет сформировать написанные в нашем модуле документ, который будет автоматически заполнен только данными учетных записей уволенных сотрудников. После проведения документа, благодаря настройкам интеграции, недействующие учетные запись будут удалены. И таких возможностей интеграции бесконечное множество.

Репкин Игорь Алексеевич	i.repkin	i.repkin@kbp.ru
Добровольская Ольга Николаевна	o.dobrovolskaya	o.dobrovolskaya@kbp.ru
Великолуг Александр Николаевич	a.velikolug	a.velikolug@kbp.ru
Черниязов Максут Аделканович	m.cherniyazov	m.cherniyazov@kbp.ru
Белокриницкая Лилия Борисовна	l.belokrinitskaya	l.belokrinitskaya@kbp.ru
Голованова Валентина Евгеньевна	v.golovanova	v.golovanova@kbp.ru
Лебедева Эвелина Витальевна	e.lebedeva	e.lebedeva@kbp.ru
Магеррамова Севда Нариман кызы	s.magerramova	s.magerramova@kbp.ru
Музыченко Лидия Николаевна	l.muzychenko	l.muzychenko@kbp.ru
Томина Юлия Григорьевна	y.tomina	y.tomina@kbp.ru
Савинкин Сергей Николаевич	s.savinkin	s.savinkin@kbp.ru
Старочкин Константин Анатольевич	k.starochkin	k.starochkin@kbp.ru
Никулина Татьяна Анатольевна	t.nikulina	t.nikulina@kbp.ru
Магомедов Максуд Гаджиевич	m.magomedov	m.magomedov@kbp.ru
Павлова Ольга Григорьевна	o.pavlova	o.pavlova@kbp.ru

Рис.2. Пример обработки для заполнения сопоставления учетных записей AD и сотрудников

Заключение

Данный модуль может использоваться как на стадии внедрения программы «1С: Предприятие», так и в момент уже продолжительного использования, что существенно упрощает работу как администраторам программ 1С, которые благодаря небольшому расширению этого модулю, могут не вести контроль за созданием и удалением пользователей внутри программы, так и системным администраторам сервера, на котором пользователи ведут свою работу, которые также могут не беспокоиться о контроле за блокировкой уволенных сотрудников. Благодаря тому, что такая интеграция имеет неограниченные возможности в расширении функционала и интеграцию возможно реализовать для любой конфигурации 1С на тех серверах, где развернута AD, этот модуль можно назвать универсальным.

Литература

1. *Новикова, Т. И.* Особенности и преимущества платформы «1С: Предприятие» / Т. И. Новикова, Ю. А. Толстикова // Актуальные проблемы авиации и космонавтики. – 2015. – Т. 1. – С. 592–593.

2. *Таран, А. Н.* Построение корпоративных сетей на основе технологий Microsoft Active Directory и Novell Edirectory: «за» и «против» / А. Н. Таран // Известия ЮФУ. Технические науки. – 2008. – Тематический выпуск. – С. 194–200.

Литвинов Даниил Сергеевич – студент 4-го курса кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: keks2013@mail.ru

Сафронов В. В. (научный руководитель) – канд. техн. наук, доц., доцент кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: vitolik@bk.ru

УДК 004.4, 004.7

АНАЛИЗ ФРЕЙМВОРКОВ SPRING BOOT И REACT JS ПРИ РАЗРАБОТКЕ КЛИЕНТ-СЕРВЕРНЫХ WEB-ПРИЛОЖЕНИЙ

М. Ю. Лихачев

Воронежский государственный университет

Введение

Одним из типов программного обеспечения, широко используемого в различных сферах, является web-приложение. Такие приложения поддерживают взаимодействие с пользователем, который с любого устройства при помощи браузера получает доступ к web-страницам и возможность отправлять запросы к серверной части. Сервер обрабатывает полученные запросы, производит необходимые вычисления и формирует новые web-страницы, которые возвращает на пользовательское устройство по протоколу HTTP (HTTPS). Данный способ организации работы реализуется технологией «клиент-сервер».

Цель настоящей работы: проанализировать одни из актуальных на сегодняшний день фреймворков: React JS для front-end разработки, и Spring Boot для back-end разработки.

С этой целью были решены следующие задачи:

1. Проведен анализ фреймворка React JS.
2. Определены преимущества и недостатки React JS в сравнении с используемыми средствами front-end разработки.
3. Проведен анализ фреймворка Spring Boot.
4. Выявлены преимущества и недостатки Spring Boot в сравнении с различными средствами back-end разработки.
5. Реализован проект «Музыкальный сервис» с использованием рассматриваемых фреймворков.

Объектом работы является изучение и анализ актуальных средств web-разработки.

Предмет работы: front-end и back-end web-разработка.

Положения, выносимые на защиту:

1. Актуальность front-end фреймворка React JS.
2. Актуальность back-end фреймворка Spring Boot.

1. Анализ предметной области

На сегодняшний день разработку web-приложений можно разделить на несколько этапов, часть которых может отсутствовать или разделена на более узкие разделы в зависимости от требований конкретного проекта.

1. Проектирование веб-приложения.
2. Разработка и создание дизайн-концепции сайта.
3. Front-end разработка:
   1) Создание макетов страниц;
   2) Создание мультимедиа и содержимого страниц;
   3) Вёрстка страниц и шаблонов.
4. Back-end разработка:

1) Программирование или интеграция в систему управления содержимым (CMS);

2) Обеспечение слаженной работы между front-end и back-end компонентами проекта.

5. Оптимизация и размещение материалов сайта.

6. Тестирование и внесение исправлений.

7. Публикация проекта и материалов на сервере.

8. Обслуживание и поддержка разработанного программного обеспечения.

2. Анализ front-end фреймворка React JS

2.1. Краткое описание

ReactJS – это библиотека JavaScript, исходный код которой был открыт Facebook в 2013 году. Этот фреймворк подходит для создания огромных веб-приложений, где данные могут меняться на регулярной основе.

2.2. Преимущества

1. Легкость изучения. React гораздо легче учится ввиду простоты его синтаксиса. Достаточно навыков написания HTML. Нет нужды в глубоком изучении TypeScript, как в случае с Angular.

2. Высокий уровень гибкости и максимальная отзывчивость.

3. Виртуальная DOM (document object model), которая позволяет упорядочивать документы форматов HTML, XHTML или XML в дерево, которое лучше всего подходит веб-браузерам для анализа различных элементов веб-приложения.

4. Легко работает при высоких нагрузках (к примеру, в сочетании с ES6/7).

5. Связывание данных от больших к меньшим. Это означает такой поток данных, при котором дочерние элементы не могут влиять на родительские данные.

6. Открытый исходный код, что приводит к множеству ежедневных обновлений и улучшений в соответствии с отзывами разработчиков.

7. Легкий вес, так как данные, которые выполняются на стороне пользователя, могут легко быть представлены на стороне сервера в то же самое время.

8. Простая миграция между версиями [1].

2.3. Недостатки

1. Нехватка официальной документации.

2. Отсутствие чёткой цели, что образует слишком большой выбор для разработчиков.

3. Долгое время для освоения, в частности требует глубокого понимания интеграции пользовательского интерфейса в структуру MVC [1].

3. Анализ back-end фреймворка Spring Boot

3.1. Краткое описание

Spring Boot – это проект, построенный по принципу convention-over-configuration (соглашения по конфигурации), призванный упросить создание приложений на базе Spring Framework [2].

3.2. Преимущества

1. Создание автономных приложений Spring.

2. Интеграция Tomcat и Jetty (без развёртывания WAR-файлов).

3. Предоставление базовых объектных моделей проектов (Project Object Models – POMs) для упрощения конфигурации Maven.

4. Автоматическая настройка Spring.

5. Предоставление готовых решений для метрик, проверки работоспособности и внешней конфигурации.

6. Устранение необходимости генерации кода и конфигурации XML [2].

3.3. Недостатки

1. Сложная кастомизация параметров приложения;

2. Не предоставляет достаточной информации для принятия обоснованного решения по выбору технологий, устранения разнообразных конфликтов и решения других часто возникающих проблем [3].

4. Реализация проекта «Музыкальный сервис»

Вышеописанные преимущества и недостатки данных фреймворков были учтены при реализации проекта «Музыкальный сервис». Его суть заключается в анализе, критике, и обсуждению авторских музыкальных произведений. Представляет из себя систему, выполняющие функции, аналогично аудиопроигрывателям, и имеют более детальную систему оценивая качества аудиофайлов.

Заключение

Результатом исследования стал анализ средств front-end и back-end разработки web-приложений. В ходе исследования были выявлены особенности вышеописанных средств разработки, были произведены сравнения с React JS и Spring Boot соответственно. На основе этого анализа выбраны необходимые средства для реализации клиент-серверного web-приложения

Литература

1. Angular 2 vs React. – Режим доступа: http://blog.techmagic.co/angular-2-vs-react-what-to-chose-in-2017 (Дата обращения: 22.04.2020).

2. Spring Boot. – Режим доступа: https://ru.bmstu.wiki/Spring_Boot (Дата обращения: 22.04.2020).

3. Стоит ли использовать Spring Boot в вашем следующем проекте?. – Режим доступа: https://habr.com/ru/post/244531 (Дата обращения: 22.04.2020).

Лихачев Максим Юрьевич – студент 4-го курса кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: max142343245@gmail.com

Горбенко Олег Данилович (научный руководитель) – канд. физ.-мат. наук, доцент, доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: oleg_dan@mail.ru

УДК 004.021

МОДИФИЦИРОВАННЫЙ АЛГОРИТМ ПОИСКА ФАКУЛЬТАТИВНЫХ ПРИЗНАКОВ ОБЪЕКТИВНОЙ СТОРОНЫ ПРЕСТУПЛЕНИЯ ПРИ КВАЛИФИКАЦИИ ПРЕСТУПНОГО ДЕЯНИЯ

М. С. Ляпина

Воронежский государственный университет

Введение

Несмотря на то, что в современном мире большая часть жизни человека компьютеризирована, всё ещё существуют целые сферы, где степень автоматизации минимальна. Одной из причин является суждение о том, что некоторые отрасли требуют непосредственного вовлечения человека в процесс. Сюда можно отнести ту деятельность, в которой требуется принятие субъективного решения. Примером может послужить юриспруденция, в частности, уголовное право, где, несмотря на то что появляется множество идей по автоматизации применения, их реального применения выявлено не было.

Поэтому была предпринята попытка автоматизации процессов разрешения конкретных правовых задач таких, как, например, определение состава преступления и выявление круга статей, подходящих для квалификации преступного деяния.

В рамках процесса квалификации преступления необходимо, во-первых, выявить все элементы состава преступления, и, во-вторых, сопоставить данный состав с составами статей УК РФ [1]. Текст правонарушения сформулирован на естественном языке, поэтому его необходимо предварительно обработать для того, чтобы сделать пригодным для дальнейшего использования, что позволит определить статьи, по которым следует квалифицировать преступление.

В ранних работах [2,3] уже было сформулировано большинство алгоритмов, цель данной – усовершенствование алгоритма поиска факультативных признаков объективной стороны преступления.

1. Теоретические сведения

Объективная сторона преступления – это один из элементов состава преступления, включающий в себя признаки, характеризующие внешнее проявление преступления в реальной действительности, доступное для наблюдения и изучения. К ним можно отнести: способ, место, время, обстановку, орудие и средства совершения преступления.

Эти признаки присущи любому преступному деянию, так как оно всегда совершается определенным способом, в конкретном месте, обстановке, в определенное время, с использованием конкретных орудий и средств, с помощью определенных приемов, влияющих в различной мере на характер и степень общественной опасности совершенного преступления. Однако по своей природе они являются факультативными, т.е. необязательными элементами составов.

В некоторых же случаях законодатель указывает в диспозиции статей Особенной части УК один или несколько из перечисленных признаков, тогда они становятся обязательными признаками состава преступления и влияют на квалификацию деяния.

2. Алгоритм поиска места и времени совершения преступления

Алгоритм поиска места и времени совершения преступления был сформулирован ранее [2, 3] и базировался на алгоритме поиска субъекта преступления.

Шаг 1. Обработка текста, представленного на естественном языке. Данный шаг был подробно разобран в предыдущих работах [2, 3] и включает сегментацию текста на предложения, токенизацию и лемматизацию.

Шаг 2. Определение общественного отношения, которое подверглось посягательству. Этот процесс так же описан в ранних работах [2, 3]. В нём выделялось слово, определяющее объект преступления.

Шаг 3. Выделение предложения, в котором находится слово, характеризующее объект преступления.

Шаг 4. Синтаксический разбор предложения. Выполняется с помощью сервиса MaltParser [4], в основе которого лежат алгоритмы машинного обучения.

Шаг 5. Поиск в предложении обстоятельства места.

Шаг 6. Поиск в предложении обстоятельства времени.

Минус данного алгоритма заключается в ненадёжности последних двух пунктов. Для более точного вычисления места и времени совершения преступления были проанализированы библиотеки, работающие с естественным языком. Лучшие результаты показала Natasha [5] – библиотека для поиска и извлечения именованных сущностей (Named-entity recognition) из текстов на русском языке, разработанная для языка Python. Поэтому последние шаги уместнее выполнить, используя данную библиотеку. Так же пункт 4 становится необязательным в данном алгоритме.

Приведём пример:

Леонов Андрей Дмитриевич систематически угрожал, унижал и избивал свою супругу Леонову Марину Алексеевну, чем довёл её до самоубийства. Женщина выпила недопустимую дозу лекарственных средств и скончалась в своей квартире по адресу ул. Большая дворянская, д. 1, 20 марта 2019 года в 20:33 по мск. времени.

Шаг 1. Данный шаг был подробно разобран в [2,3].

Шаг 2. Словоформа, определяющая объект преступления в данном случае – «скончалась».

Шаг 3. Предложение – «Женщина выпила недопустимую дозу лекарственных средств и скончалась в своей квартире по адресу ул. Большая дворянская, д. 1, 20 марта 2019 года в 20:33 по мск. времени.»

Шаг 5. Место совершения преступления – «ул. Большая дворянская, д. 1».

Шаг 6. Время совершения преступления – «20 марта 2019 года, 20:33».

3. Выбор средств реализации

Для решения многих лингвистических задач используются так называемые текстовые корпуса – специальным образом подобранные, размеченные и структурированные коллекции текстов. Поэтому и в данном случае было принято решение использовать текстовый корпус.

В результате анализа существующих корпусов русского языка, находящихся в открытом доступе не только для просмотра и поиска, но и для скачивания размеченного словаря целиком, был выбран открытый корпус русского языка OpenCorpora [7].

Ввиду того, что используя текстовый корпус, невозможно провести синтаксический анализ предложений, было принято решение использовать сторонний сервис для выполнения этой задачи.

Проанализировав имеющиеся сервисы, осуществляющие синтаксический анализ предложений, был выбран основанный на машинном обучении MaltParser [4], так как у данного инструмента есть оболочка mp4ru, которая уже предварительно обучена на текстовом корпусе СинТагРус [6] и которую можно использовать в качестве библиотеки в языке Java. К тому же, данный сервис находится в открытом доступе в интернете, лёгок в использовании и предоставляет довольно точные результаты при работе.

Для поиска адресов и даты совершения преступления была выбрана библиотека для поиска и извлечения именованных сущностей Natasha [5]. В ней собраны грамматики и словари для парсера Yargy. Natasha [5] имеет лаконичный интерфейс. Доступны экстракторы для имён, адресов, сумм денег, дат и некоторых других сущностей.

Заключение

Таким образом, алгоритм поиска места и времени совершения преступления был модифицирован и усовершенствован. В дальнейшем планируется разработать алгоритм поиска обязательного признака объективной стороны преступления. Для этого необходимо будет использовать машинное обучение, так как не представляется возможным отыскать причинно-следственную связь в тексте, используя обычные алгоритмы.

Литература

1. Уголовный кодекс Российской Федерации. Официальный текст: текст Кодекса приводится по состоянию на 07 апреля 2020 г.– Москва : Омега-Л, 2020. – 193 с.

2. *Ляпина, М. С.* Задача квалификации преступного деяния для обучающих программ / М. С. Ляпина, И. Е. Воронина // Математика, информационные технологии, приложения: сборник трудов Межвузовский научной конференции молодых ученых и студентов. – Воронеж, 2018. – С. 70–78.

3. *Ляпина, М. С.* Автоматизация принятия решения при квалификации преступного деяния / М. С. Ляпина, И. Е. Воронина // Информатика: проблемы, методология, технологии, приложения: сборник трудов XVIII Международная конференция. – Воронеж, 2018. – С. 70–78.

4. MaltParser: инструмент для работы с деревьями зависимостей – Режим доступа: http://www.maltparser.org/mco/mco.html (дата обращения: 31.05.2019).

5. Natasha библиотека для поиска и извлечения именованных сущностей зависимостей. – Режим доступа: https://github.com/natasha (дата обращения: 10.03.2020).

6. СинТагРус: национальный корпус русского языка. – Режим доступа: http://www.ruscorpora.ru/instruction-syntax.html (дата обращения: 31.05.2019).

7. OpenCorpora: национальный корпус русского языка. – Режим доступа: http://opencorpora.org/ (дата обращения: 31.05.2019).

Ляпина Мария Сергеевна – магистрант 2-го года обучения кафедры Программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: lypinam@rambler.ru

Воронина Ирина Евгеньевна (научный руководитель) – д-р тех. наук, проф., профессор кафедры программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: lypinam@rambler.ru

УДК 004.032.26

АЛГОРИТМ ДЕТЕКТИРОВАНИЯ ОБЪЕКТА ПО ОДНОМУ ОБУЧАЮЩЕМУ ПРИМЕРУ

М. С. Малик

Воронежский государственный университет

В статье рассмотрена одна из актуальных проблем нейросетевого детектирования объектов – детектирование объекта по одному обучающему примеру (англ. One-shot object detection). Всего по одному обучающему изображению, класс которого заранее не был известен на этапе обучения, необходимо обнаружить все объекты того же класса на контрольном изображении. Для решения поставленной задачи были исследованы *механизм совместного внимания и восприятия*, а также проанализирована эффективность работы алгоритма на пользовательских наборах данных.

Введение

Возможность людей учить что-то новое с ограниченным числом обучающего материала поразительна. Рассмотрим, например, задачу опознания и локализации конкретного объекта на фотографии или картинке. Человек, который раньше не видел этот объект и не знал, как он выглядел, сможет это сделать, основываясь всего на одном фото-примере. Даже без какого-либо знания об объекте, наша система обработки изображений развита достаточно хорошо, чтобы выполнять такого рода задачи посредством разных функциональных возможностей: восприятие всех пикселей объекта как единое целое, извлечение характерных черт для сравнения и механизм внимания для локализации объекта на фотографии. И все это происходит при работе с изображениями, на которых объект находится под разным освещением и при разных ракурсах.

1. Реализация алгоритма

1.1. Постановка задачи

Рассмотрим задачу детектирования объекта по набору меток класса. Обозначим набор как C. Так как метод разработан для сценария детектирования объекта по одному примеру, то разделим метки на $C = C_0 \bigcup C_1$, где C_0 – метки классов, доступные во время обучения, C_1 – метки классов, для проверки алгоритма. Была разработана сетевая архитектура на основе Faster R-CNN. Также была использована сеть ResNet-50 как основной сверточной нейронной сетью.

Формулировка задачи выглядит следующим образом: Дано изображение p, которое является экземпляром конкретного класса объекта из C_1, тогда необходимо найти все соответствующие объекты того же класса в контрольном наборе изображений I. Также, предполагается, что контрольный набор изображений включает в себя по меньшей мере один экземпляр объекта относительно искомого изображения.

1.2. Гипотезы о местоположении объекта с использованием нелокальных операций

Обозначим тренировочную выборку как D, с информацией об ограничивающих прямоугольниках (англ. bounding boxes) для меток классов из C_0. Ввиду того, что мы применяем архи-

тектуру Faster R-CNN для детектирования объекта, возникает резонный вопрос, а именно, правильно ли были сгенерированы предложение относительно регионов (англ. region proposal) на основе сети для предложения регионов – RPN (англ. region proposal network). Вспомним, что обучение RPN использует информацию о наличии ограничивающих прямоугольников над всеми классами объектов в каждом изображении. Однако в нашем случае только достоверные прямоугольники (англ. ground-truth bounding boxes), которые соответствуют меткам в C_0 могут быть использованы во время обучения RPN. Ограничение подразумевает, что если искомый класс объекта из C_1 значительно отличается от любого из C_0, то RPN может не дать ожидаемый набор гипотез для детектирования соответствующих экземпляров объекта в контрольном изображении. Для разрешения этой ситуации была улучшена сверточная карта признаков нелокальными операциями [3]. Например, пусть I – будет контрольным изображением, а p – искомым. Сверточная карта признаков, используемая RPN для генерации предложений, будет выражена следующим образом: $\varphi(I) \in R^{N \times W_1 \times H_1}$, а $\varphi(p) \in R^{N \times W_p \times H_p}$ будет отображать карту признаков p с того же сверточного слоя. Рассматривая $\varphi(p)$ как входной параметр, нелокальная операция будет применена к $\varphi(I)$ в результате чего, будет представлен нелокальный блок $\psi(I; p) \in R^{N \times W_1 \times H_1}$. Аналогично мы можем получить нелокальный блок $\psi(p; I) \in R^{N \times W_p \times H_p}$, используя $\varphi(I)$ в качестве входного параметра. Взаимные нелокальные операции между I и p на самом деле и являются механизмом взаимного внимания. В итоге мы можем представить две расширенные сверточные карты признаков в виде:

$$F(I) = \varphi(I) \oplus \psi(I; p) \in R^{N \times W_1 \times H_1}, \text{ для контрольного изображения } I,$$

$$F(p) = \varphi(p) \oplus \psi(p; I) \in R^{N \times W_p \times H_p}, \text{ для искомого изображения } p,$$

где знак $\oplus$ – поэлементная сумма по исходным картам признаков φ и нелокальному блоку ψ. Поскольку $F(I)$ состоит не только лишь из признаков целевого изображения I, но также из взвешенных (или сопряженных) признаков между I и p. Таким образом разработанная RPN на основе расширенных карт признаков будет извлекать больше информации из искомого изображения и генерировать гипотезы о предлагаемых регионах намного лучше. Другими словами, полученные нелокальные гипотезы регионов будут более подходящими для решения задачи детектирования по одному примеру.

1.3. Механизмы сжатия и совместного возбуждения

Помимо связывания генерацию гипотез предложенных регионов с изображением искомого объекта, механизм взаимного внимания, построенный на основе нелокальных операциях, приводит две карты признаков $F(I)$ и $F(p)$ к одной размерности каналов (для примера N). Связь между этими картами может быть дополнена механизмом *сжатия и совместного возбуждения SCE* (англ. squeeze-and-co-excitation) таким образом, что изображение p может стать соответствующим предложенной области (региону), с помощью пересчета весов, распределенных по N каналам. В частности, на этапе сжатия происходит суммирование каждой карты признаков относительно их размерности с помощью общей операции объединения GAP (англ. global average pooling), а *механизм совместного возбуждения* служит «мостом» между $F(I)$ и $F(p)$ для того, чтобы одновременно выделить те каналы, которые играют ключевую роль в оценки меры схожести. Между слоем сжатия и слоем совместного возбуждения находятся 2 полносвязных слоя, или, по-другому, многослойный персептрон (MLP)[1]. Операции *механизма сжатия и совместного возбуждения* могут быть представлены следующим образом:

$$SCE(F(p), F(I)) = w, \ \tilde{F}(p) = w \odot F(p), \ \tilde{F}(I) = w \odot F(I),$$

где $\tilde{F}(p)$ и $\tilde{F}(I)$ являются пересчитанными картами признаков, $w \in R^N$ – вектор совместного возбуждения и знак $\odot$ – является поэлементным произведением. С данными формулами p может быть представлено следующим образом:

$$q = w \odot GAP(F(p)) = GAP(\tilde{F}(p)) \in R^N,$$

а вектор признака, обозначим его как r, сгенерированный RPN для гипотез о предложении регионов, может быть посчитан аналогичным образом, а именно, выполнив операцию GAP над соответствующим регионом $\tilde{F}(I)$.

1.4. Рейтинг гипотез

Предположим, что K предложенных областей, сгенерированных RPN, выбраны в качестве возможных кандидатов для детектирования объектов относительно искомого изображения p. Была создана 2-х слойная MLP сеть M, в которой последний слой – это двухканальный softmax. На этапе тренировки, были промаркированы все из K предложенных областей как передний план (метка = 1) или задний план (метка = 2) исходя из значения IoU (Interception over union) с достоверными ограничивающими прямоугольниками относительно 0.5. Затем мы рассматриваем *рейтинговую ошибку отклонений* для коэффициентов для того, чтобы в неявном виде обучить желаемую метрику таким образом, что наиболее подходящие гипотезы предложения по области для p находились бы на верхних позициях в рейтинговом списке. Для этой цели мы соединяем для каждой гипотезы ее вектор признаков r с вектором признаков q для того чтобы получить общий вектор, обозначенный как $x = [r^T; q^T]^T \in R^{2N}$, у которого метка $y = 1$ если r соответствует переднему плану и $y = 0$ в противном случае. Сеть M соответственно имеет размерность $2N \to 8 \to 2$. Теперь пусть $s = M(x)$ будет вероятностью переднего плана изображения, предсказанной M относительно q. Тогда определим рейтинговою ошибку отклонений коэффициентов следующим образом:

$$L_{MR}(\{x_i\}) = \sum_{i=1}^{K} y_i \times \max\{m^+ - s_i, 0\} + (1 - y_i) \times \max\{s_i - m^-, 0\} + \Delta_i,$$

$$\Delta_i = \sum_{j=i+1}^{K} [y_i = y_j] \times \max\{|s_i - s_j| - m^-, 0\} + [y_i \neq y_j] \times \max\{m^+ - |s_i - s_j|, 0\},$$

где $[\bullet]$ скобка Айверсона, m^+ – ожидаемая вероятность нижней границы для прогнозирования приоритетной области и m^- – ожидаемая вероятность верхней границы для прогнозирования неприоритетной области. В данной реализации $m^+ = 0.7$ и $m^- = 0.3$.

В итоге, общая ошибка для архитектуры, показанной на рис. 1, выглядит следующим образом:

$$L = L_{CE} + L_{\mathrm{Re}g} + \lambda L_{MR},$$

где первые два слагаемых это функция потерь перекрестной энтропии и ошибка регрессии Faster R-CNN, а коэффициент $\lambda = 3$.

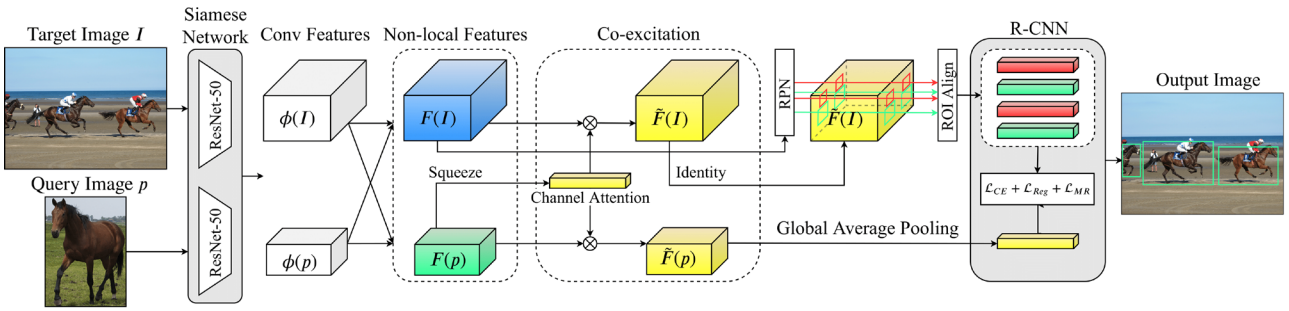


Рис. 1. Архитектура нейросетевой модели

1.5. Анализ работы алгоритма

Данный алгоритм был протестирован на 2-х популярных датасетах VOC (Visual object classes) и COCO (Common objects in context). Для тестирования использовались VOC 2007 train&val и VOC 2012 train&val иоценка проводилась на VOC 2007 test выборке. Результаты приведены в табл. 1.

Таблица 1

Метрики на датасетах VOC и COCO

Датасеты	Классы, присутствовавшие во время обучения								Классы, отсутствующие на момент обучения	
	человек	поезд	лошадь	собака	птица	стол	диван	тв	корова	кошка
VOC	55.7	55.4	77.3	80.4	70.1	43.4	51.1	60.5	71.8	70.4
COCO	64.3	58.3	76.2	78.2	69.1	46.6	50.0	62.5	72.8	63.4

Для VOC2007-2012 датасета mAP = 56.1, а для COCO mAP = 59.2.

Заключение

С помощью *механизмов совместного внимания и восприятия* возможно изменить алгоритм обучения таким образом, что он сможет опираться не только на информацию о классах в тренировочных данных. Оба механизма нацелены на то, чтобы извлечь как можно больше неявной информации в искомом и контрольном изображениях. Такого рода информация является общей и не принадлежит исключительно тренировочной выборке. Как результат, данный алгоритм может генерировать гипотезы нелокальных предложений для объектов и использовать *механизм совместного восприятия* для того, чтобы выделить важные общие признаки обоих изображений. В дальнейшем планируется обобщить алгоритм для работы на нескольких обучающих примерах.

К минусам данного алгоритма можно отнести довольно сложную архитектуру сети, а также то, что необходимо заранее обучать сверточные нейронные сети для извлечения признаков из контрольного и искомого изображений. Такая подготовка требует много времени.

Литература

1. *Gregory R. Koch*. Siamese neural networks for one-shot image recognition/ Gregory R. Koch. // University of Toronto – 2015.– С. 13–15.

2. *Jie Hu*. Squeeze-and-excitation networks / Jie Hu, Li Shen, and Gang Sun. – 2018.

3. Non-local neural networks / Xiaolong Wang [и др.] // 2018 IEEE Conference on Computer Vision and Pattern Recognition. – Salt Lake City, UT, USA,2018.

4. One-Shot Object detection with Co-Attention and Co-Excitation / Yi-Chen Lo, Hwann-Tzong Chen, Tyng-Luh Liu // Neural Information Processing Systems. – 2019.

Малик Матвей Сурендерович – магистрант 2-го года обучения кафедры математического и программного обеспечения электронных вычислительных машин Воронежского государственного университета. E-mail: malik_ms96@mail.ru.

Зонов Андрей Владимирович (научный руководитель) – преподаватель кафедры математического и программного обеспечения электронных вычислительных машин Воронежского государственного университета. E-mail: zonov@amm.vsu.ru

УДК 004.056.53

АНАЛИЗ СОВРЕМЕННЫХ МЕТОДОВ ПОИСКА АНОМАЛИЙ В СИСТЕМАХ ОБНАРУЖЕНИЯ ВТОРЖЕНИЙ

Е. Н. Нагула

Воронежский государственный университет

Введение

Все чаще эксперты по компьютерной безопасности приходят к мнению, что даже самые надежные системы защиты не способны защитить от атак компьютерные системы государственных и коммерческих учреждений. Одна из причин в том, что в большинстве систем безопасности применяют стандартные механизмы защиты: идентификацию и аутентификацию, механизмы ограничения доступа к информации в соответствии с правами субъектов и криптографические механизмы. Такой традиционный подход имеет определенные недостатки, а именно: незащищенность от собственных пользователей-злоумышленников, размытость разделения субъектов системы на «своих» и «чужих» через глобализацию информационных ресурсов, легкость в подборе паролей, в следствии использования их содержательной разновидности, снижение производительности и осложнения информационных коммуникаций в следствии ограничения доступа к ресурсам организации. Поэтому возникла потребность в механизмах, которые бы дополняли традиционные, и давали возможность выявить попытки несанкционированного доступа и информировали об этом ответственных за безопасность или реагировали в ответ.

Важным фактором является то, чтобы такие системы могли противостоять атакам, даже если злоумышленник уже был аутентифицирован и авторизован, и с формального взгляда соблюдения прав доступа имел необходимые полномочия на свои действия. Такие функции и выполняют системы обнаружения вторжений *IDS* (*intrusion detection systems*). Поскольку предусмотреть все сценарии развертывания событий в системе с активным «чужим» субъектом невозможно, нужно или подробнее описать возможные «злонамеренные» сценарии или же, наоборот, описать «нормальные» сценарии с учетом, что всякая активность, на которую не распространяется принятое понимание «нормальности», является опасной.

Системы IDS делятся на системы обнаружения злоупотреблений *MDS* (*misuse detection systems*), которые реагируют на известные атаки, и на системы обнаружения аномалий *ADS* (*anomaly detection systems*), которые регистрируют отклонения поведения системы от нормального течения.

1. Состояние проблемы

Известно, что атака является многошаговым процессом, его осуществление требует высокой квалификации злоумышленника. Поэтому самым простым способом взлома или приведения системы в недееспособное состояние является применение «*эксплойтов*», то есть уже написанных модулей, что реализуют необходимые этапы атаки. Огромное количество эксплойтов доступны через сеть Интернет, что привело к распространению именно такого вида атаки. Как следствие, часто последовательность событий, которые реализуют атаку, фиксированная. Работа MDS базируется на создании шаблонов таких атак.

Их уровень абстракции может быть простым, таким как наличие определенных значений в заголовке сетевого пакета или последовательности команд в файле аудита, или сложным, таким как прохождение траектории системы в пространстве состояний через определенные небезопасные состояния. Системы защиты такого типа эффективны, когда схема атак известна, однако в случаях неизвестной атаки или отклонений течения атаки от шаблона – возникают проблемы. Поэтому следует поддерживать большую базу данных для каждой атаки и ее вариации, а также организовывать непрерывное пополнение баз шаблонов.

Основным предположением ADS является то, что действия злоумышленника (события в атакованной системе) обязательно отличаются от поведения обычного пользователя (от событий в нормальном состоянии), то есть являются аномалиями. Поэтому такие системы способны регистрировать и неизвестные атаки. Работе ADS предшествует период накопления информации, когда складывается концепция нормальной активности системы, процесса или пользователя. Она становится эталоном, относительно которого оценивают следующие данные. Здесь определяется оптимальное количество факторов, за которыми будут вестись наблюдения. Их совокупность не должна быть слишком большой, поскольку это снизит общую производительность работы. Она также не должна быть слишком ограниченной, поскольку по недостаточно исчерпывающим характеристикам невозможно будет построить профиль нормального поведения.

Возможны два общих вида ошибок:

• нормальное поведение системы или пользователя ошибочно принимается за злонамеренное (*false positives*);

• попытка злонамеренного проникновения в систему принимается за нормальную активность (*false negatives*).

Хотя ни одна из этих ситуаций нежелательна, но вторая все же опаснее, и поэтому одной из основных задач построения ADS является четкое определение условий, при которых ситуация воспринимается как аномальная, так, чтобы ни одна из перечисленных ситуаций не возникала слишком часто.

2. Анализ методов обнаружения аномалий в компьютерных системах

Большинство попыток построения ADS – концептуальные модели, цель которых – проверить возможность применения математической модели или подхода. Коммерческих продуктов в области IDS очень мало, а те, что есть, почти никогда не выходят за пределы MDS. Практически все описанные в литературе методы для выявления аномалий можно разделить на:

• базирующиеся на хранении примеров поведения;
• частотные;
• нейросетевые;
• строящие конечные автоматы;
• иные специальные методы.

2.1. Методы, основанные на хранении примеров поведения

Самым простым подходом является прямое запоминание примеров действий, последовательностей команд пользователей или вообще любых параметров, доступных при регистрации (*instance-based learning*). Несмотря на невозможность применения этого подхода в других случаях моделирования человеческого поведения, в задачах обнаружения вторжений он является довольно действенным. Данный факт обусловлен ограниченным количеством возможных действий субъектов компьютерной системы, значительной детерминированностью задач

и самой структурой операционной системы. Реакцией компьютерной системы на аномальное поведение процесса является его принудительное замедление. Таким образом, процессы, что демонстрируют быстрое аномальное поведение, будут почти остановлены и автоматически уничтожены системой. Зафиксированные ранее подпоследовательности системных вызовов, что входили в тренировочное множество, запоминаются, и во время работы проверяется их наличие в текущей сессии. Поскольку наблюдение ведется по системным вызовам программы, то через их высокую регулярность размеры базы подпоследовательностей не будут большими. Подпоследовательность с текущей сессии, которой не было среди тренировочных, считается аномальной. Подход требует доработки ядра операционной системы, что нелегко и не всегда возможно. Кроме того, постоянное наличие такого мониторингового компонента приводит к общему замедлению работы всей системы, которое составляет от 4 до 50 процентов.

Другим образцом instance-based системы является метод, в котором вводится специальная метрика сходства символьных последовательностей. Через необходимость сохранения большого количества данных на каждого пользователя возникает потребность в применении специальных методов уменьшения объема баз последовательностей. Самыми популярными являются два следующих метода:

• выборочная селекция примеров (сохраняются не все примеры последовательностей, а только последние n или все, кроме n с наименьшими вероятностями);

• преобразования базы на базу представителей классов элементов.

Через высокие требования к ресурсам instance-based системы могут применяться только в задачах обнаружения аномалий с незначительным количеством возможных сигналов и преимущественно с статическим поведением субъектов наблюдения.

2.2. Метод на базе частотной модели

Развитием идей instance-based систем является учет частотного распределения параметров системы. Ранее предлагалось хранить информацию о субъектах в шаблонах активности, выраженных в статистических терминах наборов характеристик поведения субъектов относительно определенного объекта, таких как:

• вхождение в системы;

• запуски программ;

• доступы к файлам и устройствам, с целью регистрации отклонений.

Затем проверяется попадает ли относительное количество определенных событий в заданный экспертом интервал. Модификацией частотного подхода является работа, где предлагается метод, основанный на так называемых структурных нулях. Он состоит в использовании информации о командах, которые используются очень редко или совсем не используются, и соответствующие им ячейки в таблице вероятностей равны нулю, то есть является структурными нулями. Вводится индекс уникальности, что исчисляется для каждой сессии и каждого пользователя. Этот индекс получает положительный прирост для частых команд в рамках текущей сессии, который тем меньше, чем чаще эта команда используется. Так, широкое использование жидких команд повлечет большие значения индекса уникальности. В случае возникновения команд, не присущих пользователю, индекс уменьшается. Предположив, что значение индекса является стабильным для определенного субъекта, аномалии пытаются различать по значениям индекса. Распространенными недостатками частотных методов является неадаптивность, поскольку часто эталонные значения частот определяются однократно, за тренировочным множеством или по экспертным данным, и не учитывается последовательности выполнения команд.

2.3. Метод на базе нейросетевой модели

Применение нейронных сетей обусловлено самой неформальной постановкой задачи – выявить аномальное поведение. Идея заключается в том, чтобы, получив некоторое «тренировочное» множество параметров вход-выход, что характеризует поведение системы, дать сети «привыкнуть» к ним. Выходом может быть некоторый коэффициент «нормальности» поведения или один из параметров системы. Если входные данные имеют закономерности, то делают предположение, что сеть способна «обучаться» на них. Если в процессе работы предложенный нейронной сетью выход является некоторым коэффициентом, попадающим в опасную область или отличающийся от имеющегося в реальной системе, то делается вывод, что в системе имеется аномалия. Для построения шаблона поведения пользователя используются такие параметры как:

- часы, в которые он обычно работает;
- набор узлов, с которых он начинает рабочую сессию;
- характеристики использования ресурсов системы.

Эти параметры оцифровываются и являются входными для нейронной сети обратного распространения ошибки (*backpropagation neural network, BPNN*), а выходным является коэффициент, что составляет ноль для пользователя с нормальным поведением и единица – с аномальной.

Поскольку для получения «ненормального» поведения надо было бы заставить пользователя обращаться не так, как он привык, то аномальные данные генерируются случайно, что затрудняет интерпретацию результатов относительно работы с реальными данными. Результаты исследований данных методов свидетельствуют о возможности применения такого подхода и об эффективности выбранного метода кодирования команд. Однако нерегулярность поведения пользователей значительно повышает уровень ошибок типа false positives. Ожидается, что лучшие результаты можно получить на данных от регулярных источников, таких как системные процессы. Неисследованной остается возможность применения нейронных сетей неперсептронного типа, например, ансамблевых. Недостатком многих нейронных сетей является их плохая приспособленность для работы с неупорядоченными величинами. Введение искусственного порядка на множестве значений элементов только исказит картину, поскольку нейронная сеть учитывает близость числовых величин.

2.4. Метод, базирующийся на моделях, которые строят конечные автоматы

В этом методе достигается большая моделирующая способность, чем в случае использования тривиальных частотных и instance-based методов. Входные данные рассматриваются как поток дискретных событий, например, системных вызовов или идентификаторов процесса. Цель – получить автомат, который моделирует указанную последовательность событий.

Для многих последовательностей характерно, что вероятность следующего символа, элемента или сигнала, зависит от предыдущих. Часто они зависят только от небольшого количества предыдущих. Это наталкивает на мысль моделировать их с *помощью марковских цепей*. Однако в случае роста порядка цепей, что может существенно увеличить точность модели, количество состояний соответствующего автомата ведет себя как $O(\sum L)$, где $\sum$ – размер алфавита символов; L – порядок цепи.

Это ставит большие требования к ресурсам и увеличивает время обработки. По входным данным строится матрица переходов цепи первого порядка и вероятность сессии определяется как произведение вероятностей перехода между состояниями, что соответствуют элемен-

тарным событиям в файле аудита. Мощной моделью являются *скрытые марковские модели* (*СММ*). От марковских цепей они отличаются тем, что выходные символы автомата не детерминированы его состояниями, а зависят от них стохастически. Кроме того, часто необходим подбор количества скрытых состояний, который обычно проводится вручную и после тренировки реально задействованными оказываются лишь некоторые из них. Для обеспечения адаптивности необходимо периодически перенастраивать СММ на новых данных, что является значительным недостатком модели.

Преимуществом такого подхода является отсутствие ошибок типа false positives из-за зависимости построенных моделей непосредственно от исходного кода программы. То есть несовместимая с моделью последовательность вызовов сразу сигнализирует об отклонениях работы программы от ее исходного кода, и только такие отклонения сигнализируют об аномалии. Соответственно, все, что предусмотрено текстом программы, будет принято. Основной недостаток такого подхода – сложный процесс построения модели по анализу текстов программы. Кроме того, есть ограничения на виды программ, которые можно описать статическими грамматиками.

Существует модель, которой не свойственны недостатки СММ и марковских цепей с фиксированным порядком. Ее основная идея – использовать модель марковских цепей переменного порядка, поскольку следующий символ или сигнал в последовательностях, что генерируются в компьютерных системах, редко определяется предыдущим контекстом постоянной длины. Учет только существенных контекстов дает возможность обойти экспоненциальный рост требований к ресурсам, не ухудшая точности модели. Кроме того, большинство предыдущих подходов к моделированию поведения являются неадаптивными. Лучшее, что предлагается – периодическая перенастройка модели. Отсюда потери времени и уменьшение надежности систем в результате очередной перетренировки. Для поддержки реальной адаптивности применяется определенная процедура изменения параметров модели так, что последняя полученная информация становится весомой. При этом сохраняются аппроксимирующие свойства исходной модели.

2.5. Другие методы

Кроме описанных выше методов, для определения аномалий используются Байесовские сети – графические модели представления в собственной структуре зависимостей между объектами и распределением вероятностей. Оценивается корреляция между состояниями, а также строится сеть с соединенными зависимыми между собой узлами и вероятностями переходов между состояниями. Вероятности связаны с узлами таблицы распределения вероятностей по данным состояний предыдущих узлов. Через жесткую зависимость построенной структуры от предоставленных тренировочных данных полученная модель не является адаптивной.

Существуют попытки использовать для задачи обнаружения аномалий генетические алгоритмы и алгоритмы поиска оптимума, что базируются на аналогиях с естественным отбором в природе. Вместо «популяции» берется множество «особей» – бинарных строк фиксированной длины. Количество строк в множестве также фиксированная величина. В процессе эволюции, по определенным критериям «приспособленности к окружающей среды» выбираются строки, на базе которых генерируется следующее поколение строк. Генерация следующего поколения происходит с использованием трех основных операторов над строками:

- селекция (выбор наиболее приспособленных представителя);
- репродукция (обмен частями строк между собой);
- мутация (случайные изменения битов в строках для предотвращения необратимой потери информации).

Генетические алгоритмы используются для нахождения пессимистического сценария в гибридной задаче выявления злоупотреблений и аномалий. С одной стороны, остаются только известные атаки, с другой – возникает возможность применения генетических алгоритмов и нестрогих ограничений на гипотезы.

3. Альтернативные пути решения проблемы

Выше рассмотрены основные методы, которые применяются в ADS, однако ни один из них не гарантирует выявление всех атак. Итак, в реальных IDS, опираясь на анализ вышеуказанных методов, можно порекомендовать реализовывать комбинацию различных методов и на ее основании делать окончательный вывод о наличии или отсутствии вторжений, а также их характер.

Модели в этих методах не должны зависеть от конкретного типа аудит-данных, тогда их можно применить к любым аудит-последовательностям. Такими данными, кроме команд, что непосредственно выполняет пользователь, могут стать:

- последовательности системных вызовов, характерные для программы;
- значение интервалов между нажатиями клавиш на клавиатуре;
- интервалы между рабочими сессиями;
- последовательные значения координат положения курсора мыши на экране;
- последовательности значений полей заголовка сетевых пакетов всех уровней.

То есть любая последовательность сигналов, команд, элементов или других значений, характерная для пользователя, процесса, системы или сетевого сегмента, может быть использована в ADS.

Очевидно, что пользователям свойственно менять свое поведение (через изменение задач, приобретение новых привычек), поэтому модели методов обязательно должны быть адаптивными. Большая часть атак – это атаки против системных программ, например, приобретения прав суперпользователя путем использования уязвимостей в SUID программ. После этого программа, как правило, демонстрирует кардинально отличный характер системных вызовов от тех, что наблюдались до атаки. Поэтому следует обратить внимание на выявление аномалий на уровне системных вызовов, которые становятся все более популярными, поскольку дают возможность абстрагироваться от нерегулярного человеческого поведения. Одновременно нельзя отказываться от анализа аудит-данных, что поступают от пользователей, поскольку только анализ на этом уровне дает возможность выявить вторжения, которые на уровне системных вызовов не проявляются (например, использование украденного пароля).

Полноценная IDS должна содержать также компоненты, выявляющие злоупотребления, поскольку сегодня атаки с использованием эксплойтов остаются одними из основных видов атак.

Заключение

В работе рассмотрены проблемы выявления атак в компьютерных системах, системы обнаружения аномалий и присущие им наиболее известные методы. Проанализированы преимущества и недостатки каждого метода, а также предложены альтернативные пути для создания эффективной системы выявления аномалий в компьютерных системах.

Литература

1. *Axelsson, S.* The base-rate fallacy and its implications for the difficulty of intrusion detection / S. Axelsson // ACM Conference on Computer and Communications Security, 2014. – С. 1–7.

2. *Denning, D.* Anintrusion-detection model / D. Denning // IEEE Symposium on Security and Privacy, 2013. – С. 118–131.

3. *Somayaji, A.* Automated response using system-call delays / A. Somayaji // USENIX Security Symposium, 2013. – С. 185–197.

4. *Большев, А. К.* Применение нейронных сетей для обнаружения вторжений в компьютерные сети / А. К. Большев, В. В. Яновский // Вестник Санкт-Петербургского университета. – СПб., 2009. – №4. – С. 38–44.

5. Lane, T. Temporal sequence learning and data reduction for anomaly detection / T. Lane, C. E. Brodley // ACM Transactions on Information and System Security, 2008. – С. 295–331.

6. *Theus, M.* Intrusion detection based on structural zeroes / M. Theus, M. Schonlau // StatisticalComputing and Graphics Newsletter, 2014. – С. 12–17.

7. *Tan, K.* The application of neural networks to unix computer security / K. Tan // IEEE International Conference on Neural Networks, 2016. – С. 476–481.

8. *Резник, А. М.* Нейросетевая идентификация поведения пользователей компьютерных систем / А. М. Резник, Н. Н. Куссуль, А. М. Соколов // Кибернетика и вычислительная техника. – 2010. – № 123. – С. 70–79.

9. *Kevin, S.* Comparing Anomaly-Detection Algorithms for Keystroke Dynamics / S. Kevin // IEEE, 2009. – С. 125–134.

10. *Eskin, E.* Anomaly detection over noisy data using learned probability distributions / E. Eslkin // 17th International Conf. on Machine Learning, 2010. – С. 255–262.

Нагула Елена Николаевна – магистрант 2-го курса кафедры МОЭВМ Воронежского государственного университета. E-mail: nagula.helen@gmail.com

Борисенков Дмитрий Васильевич (научный руководитель) – канд. техн. наук, доцент кафедры МО ЭВМ, Воронежского государственного университета. E-mail: xuser@relex.ru

УДК 519.8

РАЗРАБОТКА ПРОЦЕДУРЫ ОЦЕНИВАНИЯ ПОРТФОЛИО ДЛЯ ВОЗМОЖНОСТИ ИСПОЛЬЗОВАНИЯ В РАБОТЕ ОБРАЗОВАТЕЛЬНОЙ ОРГАНИЗАЦИИ

С. А. Палкина, В. В. Ухлова

Воронежский государственный университет

Введение

На сегодняшний день одним из способов представления результатов образовательной и любой иной деятельности учащегося является портфолио. Материалы, содержащиеся в нем, представляют собой документы, подтверждающие образовательную и внеурочную деятельность учащегося. Портфолио уже прекрасно зарекомендовало себя как инструмент для оценки достижений учащихся в целом, так и для рейтингования при отборе для различных проектов. Однако использование портфолио в качестве дополнительного документа при поступлении в ВУЗы до сих пор затруднено. Связано это, в первую очередь, с отсутствием единой системы оценивания достижений учащегося, отображаемых в портфолио. Как показывает практика, каждое учебное заведение разрабатывает собственную шкалу и использует ее для оценивания портфолио. При этом получаемый итоговый балл характеризует оценку портфолио только в системе критериев своей образовательной организации. В то время как ВУЗ должен подойти в оценивании объективно ко всем портфолио. Более того, шкала, применяемая в общеобразовательных учреждениях, предусматривает оценивание лишь общего уровня учащегося, в то время как по нему нельзя судить о способностях (достижениях) в каком-то определенном виде деятельности (творческой, спортивной и т.п.). Существование этой проблемы и делает работу актуальной.

1. Постановка задачи

1.1. Объект исследования

Портфолио учащегося представляет собой набор документов, подтверждающих достижения в урочной и внеурочной деятельности. Среди них обязательными являются: представление информации об образовательной траектории учащегося, ведомость оценок, результаты диагностических исследований. Портфолио формируется учащимся самостоятельно и на добровольной основе. Поскольку оно может включать в себя большое количество материалов, рекомендовано организовывать тома по годам или за некий ограниченный период, например, 10–11 класс. Содержание портфолио утверждается отдельно каждой учебной организацией. В табл. 1 представлено типовое содержание портфолио учащегося МБОУ «Лицей «МОК 2».

1.2. Существующее решение

Для образовательной организации портфолио является одним из инструментов формирования рейтинга учащихся при отборе в 10 классы или на какие-либо проекты. Для этого по каждому портфолио вычисляется итоговый балл.

Итоговый балл портфолио формируется на основании следующих показателей:

1) учебная деятельность;

2) независимая оценка достижений;

Таблица 1

Содержание портфолио

№ п.п	Раздел	Стр.
1	Титульный лист	1
2	Образовательная траектория учащегося за год (посещаемые занятия, время занятий, Ф.И.О. руководителя): – учебные курсы по выбору учащегося в рамках обязательной учебной нагрузки; – внеурочные занятия в школе; – дополнительное образование в школе; – дополнительное образование вне школы.	2
3	Портфолио достижений: – ведомость отметок; – результаты диагностических исследований по учебной деятельности.	3
4	Успешность: – учебная деятельность (олимпиады, интеллектуальные предметные соревнования); – участие в проектной деятельности, связанной с процессом обучения (конференции, НОУ и т.п.); – публикации научно–исследовательского характера; – участие в интеллектуальных мероприятиях, образовательных событиях; – социальная и общественно–значимая деятельность (выступления на концертах, фестивалях); – участие в спортивных мероприятиях; – участие в творческих мероприятиях.	5
5	Портфолио работ: – лучшие учебные работы; – лучшие творческие работы.	7

3) внеурочная предметная деятельность в школе;
4) достижения в олимпиадах, интеллектуальных конкурсах, конференциях, публикации;
5) другие достижения в интеллектуально-познавательной деятельности;
6) участие в общественной жизни;
7) спортивные достижения;
8) творческие достижения;
9) отзывы;
10) рефлексия;
11) творческое оформление портфолио.

Показатели 1–9 рассчитываются на основании документов (подтвержденной информации) предоставленной учеником, администрацией школы и сторонними организациями. Показатели 10 и 11 являются оценкой независимых экспертов.

Портфолио с большим итоговым баллом в рейтинге занимает более высокую позицию, чем портфолио с меньшим значением. При равенстве баллов, более высокий рейтинг определяется сравнением отдельных показателей портфолио. Выбор значимого показателя определятся экспертом, осуществляющим рейтингование.

1.3. Цель исследования

Согласно применяемой общеобразовательными организациями методики расчета итогового балла, шкала оценивания материалов выбирается самостоятельно. В рамках одной организации такой подход может быть обоснованным. Однако с позиции ВУЗов, он не является объективным. Это связано с тем, что ряд показателей для школы и ВУЗа могут иметь различную степень важности. Более того, ВУЗ должен оценивать все портфолио в одной и той же системе. Наличие данной проблемы определяет актуальность исследования. Целью, которого, является разработка процедуры оценивания портфолио учащихся, позволяющего использовать его как в общеобразовательных организациях, так и ВУЗах.

2. Процедура оценивания материалов портфолио

2.1. Модель оценивания материалов портфолио

Процедура оценивания материалов портфолио заключается в вычислении итогового балла портфолио. Итоговый балл складывается из оценок отдельных материалов портфолио. Оценка материала портфолио определяется посредством применения шкалы оценивания к критериям (характеристикам) материалов. Соответственно, для оценивания материалов портфолио требуется формирование модели оценивания материалов, которая позволит использовать портфолио как в общеобразовательной организации, так и ВУЗе.

В подходе, изложенном в данной работе, в качестве критериев оценивания взяты основные показатели портфолио общеобразовательной организации, приведенные в табл.1. Использование метода анализа иерархии (МАИ) позволяет при формировании модели учесть несколько критериев одновременно, при этом использовать как количественные, так и качественные оценки. Более того, в ней нет ограничений на виды документов, подтверждающих достижения. Полученная модель является иерархической (рис. 1).

Первый уровень иерархии определяет первоначальный критерий отбора – вид деятельности. В данной модели выделены спортивная, образовательная, творческая и общественная деятельность. Второй уровень – виды возможной активности в каждой деятельности. Активность в данность модели рассматривается как траектория или карьерная лестница, участие в мероприятиях, достижения и отзывы о деятельности кандидата. Третий уровень – это характеристики, по которым каждую из активностей можно оценить. Так карьеру можно оценить по наличию и количеству учебных курсов, внеурочных занятий в школе и вне ее, получению дополнительного образования. Участие оценивается по виду мероприятия, его уровню значимости и форме проведения. Достижения оцениваются в зависимости от полученного при оценке балла, занятого места и присвоенной квалификации. Значимость отзывов зависит от того, кто их дал (школьные, сторонние от родственников).

Полученная модель оценки включает только основные критерии, которые в большинстве вариантов, присутствуют в любом портфолио. Это и делает ее универсальной. Она позволяет учитывать не только несколько критериев отбора одновременно, но и степень значимости отдельных из них.

2.2. Формирование шкалы оценивания

Согласно данной модели оценивания экспертом формируется портрет идеального учащегося (кандидата в какой-либо проект). Процедура формирования видения эксперта заключается в оценке значимости отдельных критериев оценивания портфолио. Для проведения процедуры оценивания значимости критериев эксперту предлагается заполнить опросный

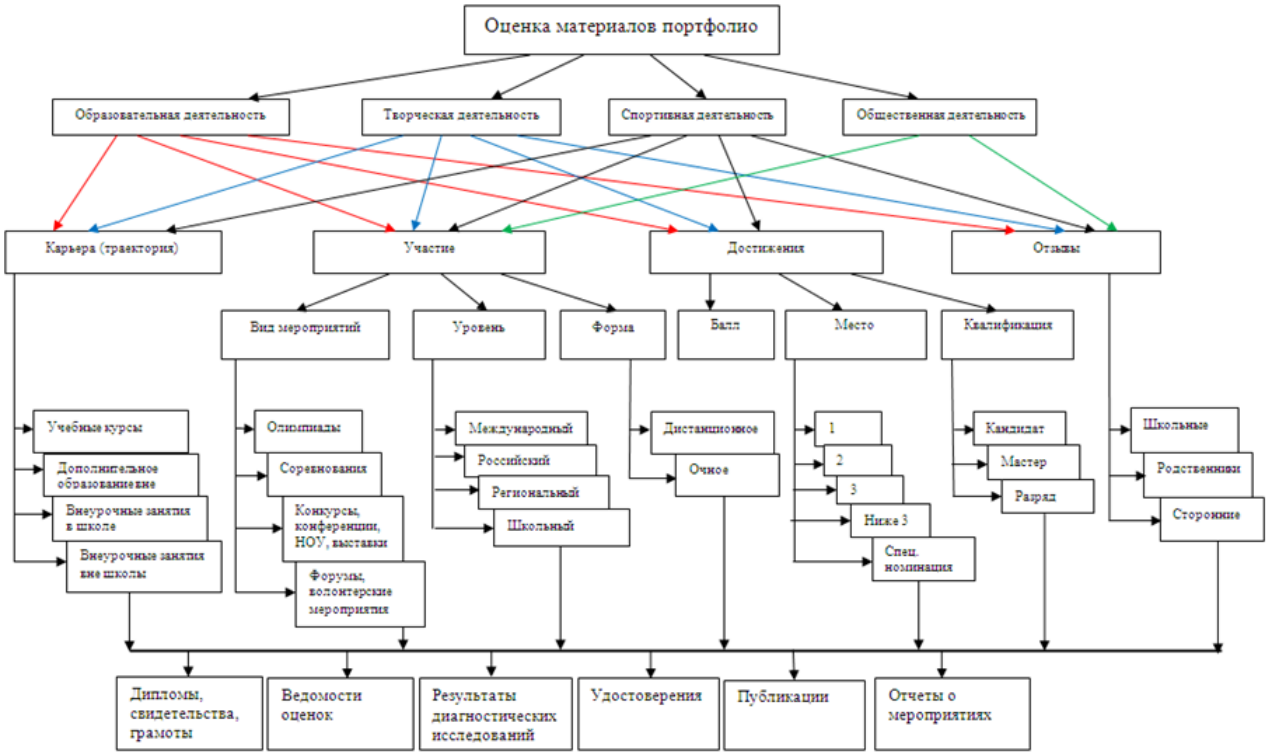


Рис. 1. Модель оценки материалов портфолио

лист. В нем эксперту предлагает расставить приоритеты критериев по каждому уровню в отдельности. Согласно используемому методу анализа иерархий по расставленным критериям, составляются матрицы парных сравнений.

Применяется следующее правило: если в цепочке приоритетов критерии стоят:

1) друг за другом, то вес ставится равным 3,

2) через один, то вес ставится равным 5,

3) через два элемента, вес ставится равным 7,

4) через три элемента, вес равен 9.

Веса значимости выбраны в соответствии со шкалой Т. Саати, применяемой для МАИ. Далее рассчитываются локальные веса критериев и на основании них глобальные. Множество значений глобальных весов представляют собой шкалу оценивания материалов портфолио. Ее значения применяются к каждому представленному в портфолио материалу.

2.3. Алгоритм проведения процедуры оценивания портфолио

Полученная шкала применяется к оцениванию всех материалов портфолио. В зависимости от задачи общеобразовательной организации количество видов материалов может изменяться. В случае, если не все критерии важны, они в цепочке оценивания ставятся. Для получения рейтинга независимо от вида деятельности при оценивании вида деятельности можно поставить всем один и тот же вес. Процедура оценивания портфолио и дальнейшего рейтингования представлена на рис. 2 в виде пошагового алгоритма.

Заключение

Разработанная процедура оценивания позволяет использовать единый подход к оценки портфолио из различных общеобразовательных организаций. При этом каждый ВУЗ может

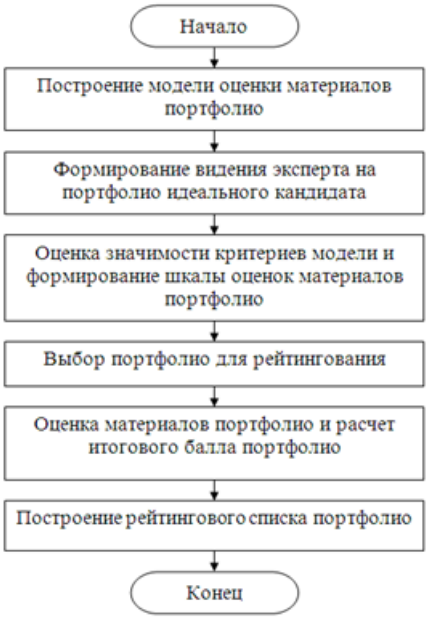


Рис. 2. Алгоритм процедуры оценивания и рейтингования портфолио

формировать собственную шкалу оценивания, которая учтет именно его критерии отбора. К ограничениям модели следует отнести набор критериев и распределение их по уровням, что может не совпасть с мнением отдельных экспертов, участвующих в формировании шкалы оценивания. В рамках отдельно взятой образовательной организации (конкретного ВУЗа) модель может быть расширена. Программная реализация предложенного алгоритма позволит автоматизировать процесс вычисления значений шкалы и, тем самым, сделать процедуру удобной для применения на практике.

Литература

1. *Саати, Т.* Принятие решений. Метод анализа иерархий / Т. Саати., пер. с англ. – Москва : Радио и связь, 1993 г. – 278 с.

2. *Макаров, И. М.* Теория выбора и принятия решений / И. М. Макаров [и др.] – Москва : Наука. 1982. – 330 с.

3. *Азарнова, Т. В.* Метод анализа иерархий как средство поддержки принятия решений в стратегическом аналитическом планировании / Т. В. Азарнова, О. Ю. Пономарева, В. В. Ухлова // Экономическое прогнозирование: модели и методы: сб. тр. IX международной науч.-практ. конф. (Воронеж, 26 апреля 2013 г.). – Воронеж, 2013. – С. 9–12.

4. Положение о портфолио обучающегося Муниципального бюджетного общеобразовательного учреждения МБОУ «Лицей «МОК № 2» г. Воронежа : электронный ресурс. – Режим доступа: http://mok2.vrn.ru/doc/proekt/polozhenie%20portfolio.pdf.

Палкина София Александровна – студентка 1-го курса факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: mail.soffya@mail.ru

Ухлова Вера Владимировна (научный руководитель) – канд. физ.-мат. наук, доц. кафедры математических методов исследования операций Воронежского государственного университета. E-mail: it_content@list.ru

УДК 519.8

ОДИН ИЗ ПОДХОДОВ МОДЕЛИРОВАНИЯ ПРОЦЕССОВ РАСПРОСТРАНЕНИЯ ВИРУСНОЙ ИНФЕКЦИИ COVID-19

С. А. Палкина, В. В. Ухлова

Воронежский государственный университет

Введение

На сегодняшний день исследование процессов распространения вирусной инфекции, именуемой как Covid-19, связано с построением математических моделей для возможности прогнозирования количества заразившихся (заболевших) и сроков распространения заболеваемости. В их основу положен стохастический характер наблюдающихся процессов и в качестве базового параметра положено число инфицированных на одного уже существующего инфицированного [6]. Однако успешность борьбы с вирусной инфекцией следует рассматривать не столько с позиции прогнозирования времени распространения и интенсивности потоков, сколько с позиции возможности правильного реагирования (возможности управления) на процесс распространения инфекции. В свою очередь, это зависит от возможности обеспечения оперативной оценки состояния процесса. Это и определило цель исследования, в котором ставится задача оценки стояния процесса распространения инфекции, с целью прогнозирования возможности успешного реагирования на него. При этом под возможностью успешного реагирования следует понимать возможность контроля процесса.

1. Объект, предмет и методы исследования

Объект исследования – процесс распространения вирусной инфекции Covid-19 среди населения. Предмет исследования – модели, методы и алгоритмы оптимизации временных характеристик функционирования сложных стохастических систем с сетевой топологией. В качестве методов исследования используются методы системного анализа, теории сложных систем, теории массового обслуживания и теории графов.

Вирусная инфекция Covid-19 – потенциально тяжёлая острая респираторная инфекция, вызываемая коронавирусом SARS-CoV-2 (2019-nCoV). Представляет собой опасное заболевание, которое может протекать как в форме острой респираторной вирусной инфекции лёгкого течения, так и в тяжёлой форме, специфические осложнения которой могут включать вирусную пневмонию, Распространяется вирус воздушно-капельным путём через вдыхание распылённых в воздухе в процессе кашля или чихания капель с вирусом, а также через попадание вируса на поверхности с последующим занесением в глаза, нос или рот [7]. Таким образом, ограничений на круг возможных инфицированных и переносчиков нет.

2. Моделирование процесса распространения инфекции

2.1. Общий процесс распространения инфекции

Сопоставление очагов заболеваемости инфекции с населенными пунктами, а путей распространения инфекции со связями (транспортным сообщением) между этими пунктами, позволяет моделировать процесс распространения вирусной инфекции Covid-19 как некий сете-

вой процесс. Наличие взаимосвязи (взаимовлияния) отдельных пунктов друг на друга определяет множество всех пунктов как систему с сетевой топологией. Населенные пункты (регионы) являются узлами, а пути распространения – дуги (связи) между ними. Рассматриваемые системы являются сложными и в плане структуры, и в плане взаимосвязи между элементами. Сложность структуры выражается в достаточно большом количестве составляющих элементов и сложности топологии в их взаимосвязи (рис. 1) [1, 4]. В свою очередь, схема протекания процессов распространения инфекции представима в виде ориентированного графа $G(V, E)$, где V – множество узлов, а E – множество дуг. Каждому ребру графа может быть приписана неотрицательная величина потока, протекающего по нему, которая сопоставима с интенсивностью потока инфицированных лиц. Направление потоков задается направлением перемещения инфицированных лиц, что позволяет узлы помечать как исток и сток (для отдельно взятых потоков).

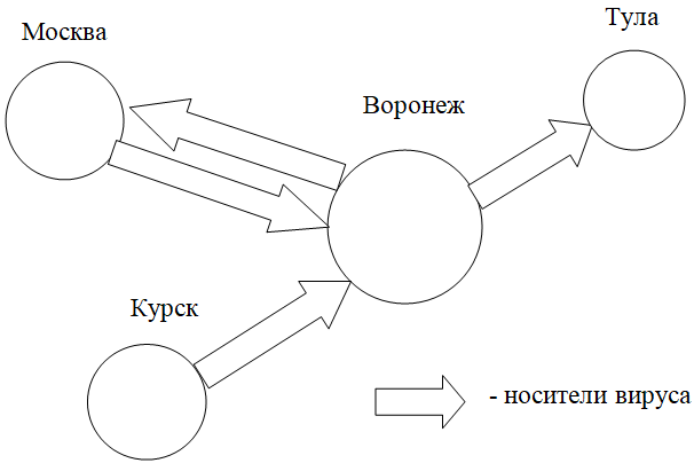


Рис. 1. Схема распространения инфекционного заболевания

Процессы, протекающие в каждом узле, это процессы возникновения источника инфекции, заражения, выздоровления и смерти населения. При этом каждый регион (населенный пункт) можно рассматривать также как отдельную систему, для которой характерно наличие этих же процессов. Более того, процессы, в такой системе (регионе) можно разделить на две большие группы: процессы, основанные на взаимодействии узла с внешней средой и внутренние процессы. В привязке к процессу распространения вирусной инфекции, процессы, связанные с внешней средой – это процессы занесения инфекции из другого региона. Внутренние процессы – это процессы, связанные с распространением инфекции внутри региона, изолированием зараженных, их выздоровлением и смертью. Такая формализация процессов узла позволяет выделить в нем ядро и пограничный слой (рис. 2).

Основными процессами пограничного слоя являются получение нового источника инфекции из внешней среды, заражение других и изоляция зараженных, переход источника инфекции в другой узел. Иными словами, существование инфицированного в населенном пункте (регионе) до попадания в медицинское учреждение определяется процессами пограничного слоя. Для ядра системы характерны процессы, описывающие процесс изоляции инфицированных и их лечение (выздоровление). Процессы смерти имеют место быть, как в пограничном слое, так и ядре системы. Процесс идентификации инфицированных, как самостоятельный процесс, в задаче не рассматривается. Следствием его существования являются процесс возникновения инфицированных в ядре системы, т.е. появление нового пациента в медицинском учреждение – это результат того, что осуществлен процесс идентификации инфицированного.

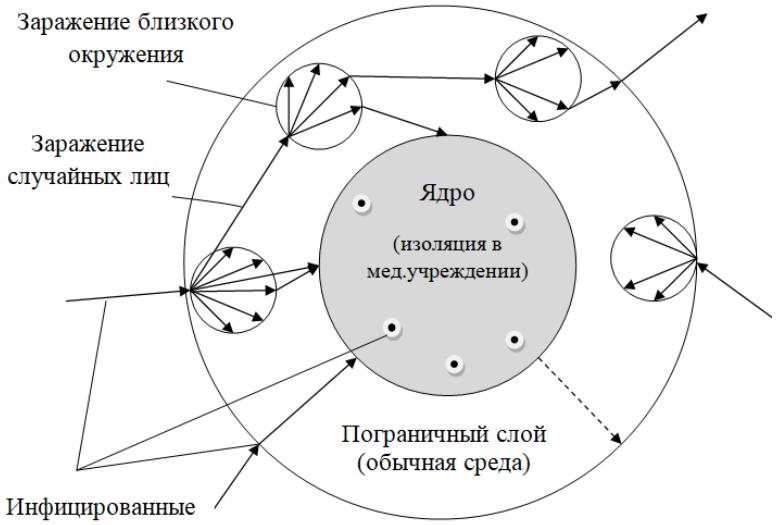


Рис. 2. Процессы узла системы (региона)

2.2. Процессы распространения инфекции в регионе: пограничный слой

С позиции математического моделирования каждый узел сети (регион или отдельный населенный пункт) можно рассматривать как систему массового обслуживания (СМО). Основным процессом в ней является «обработка, обслуживание» лиц с инфекцией. Появление в системе каждого инфицированного будем отождествлять с требованием на его обслуживание. Время, затраченное на обслуживание, будет составлять время излечения (время, когда лицо может быть опасно для населения). Потоки инфицированных лиц характеризуются интенсивностью и типом. Учитывая, что лица могут быть с подозрением на инфекцию и действительно инфицированные, следует выделять два типа таких потоков. В зависимости от типа, потоки могут иметь различные пути следования. Инфицированные – это лица, у которых подтверждено наличие инфекции. Такие лица могут либо оставаться в обычной среде, либо могут перемещаться на обсервацию в медицинское учреждение (далее ядро). Процесс обслуживания требования в пограничном слое следует уточнить. Он включает в себя процессы инфицирования ближайшего окружения и случайного населения, процесс транспортировки в ядро (помещение в медучреждение), процессы выздоровления и смерти. Опишем каждый из этих процессов в терминологии СМО.

Процесс поступления требований (инфицированных) в компонент пограничного слоя системы является случайным процессом СМО. Это определяется как случайностью поступления в систему, так и случайным характером длительности обслуживания, а соответственно, и временем пребывания требования в СМО. Такие процессы определяются в теории массового обслуживания как случайные процессы. Независимость будущего состояния процесса от прошлого, определяет этот процесс как марковский случайный процесс [1, 4, 8]. Предполагается, что входящие потоки требований являются простейшими. Простейший поток является стационарным, ординарным и обладает свойством отсутствия последействия. Интенсивностью этого потока будем считать число инфицированных, поступающих в регион за некий (определяемый) период времени. Принимая во внимание, что в привязке к рис. 1 входящих потоков может быть несколько, для дальнейшего моделирования процессов удобнее рассматривать общий поток. Характеристики объединенного потока зависят только от характеристик объединяемых потоков. Поскольку каждый поток стационарен, то общий поток так же стационарен. Общий входящий поток СМО также будет являться и простейшим [8].

Процесс обслуживания в системе, как сказано выше, включает в себя процесс заражения: как близкого окружения, так и случайных лиц. Интенсивность потоков, генерируемых этими процессами, зависит от уровня самоизолированности населения в регионе и круга контактных лиц инфицированного. Процесс заражения случайных лиц с одной стороны можно рассматривать как случайный процесс, когда он характеризуется стационарностью, ординарностью и отсутствием последействия, с другой – как самоподобный. Для самоподобных потоков характерно, что на тех или иных промежутках времени поступление инфицированного влияет на вероятности возникновения новых инфицированных и на других промежутках времени. Соответственно, анализ механизма заражения (если в зоне нахождения здорового лица есть инфицированный, то вероятность заразиться имеется) позволяет описывать поток заражения как самоподобный. Его интенсивность зависит от коэффициента подобия или индекса Херши (H). Данный индекс определяется из следующей формулы: $H = 1 - (\beta / 2), \quad 0 < \beta < 1$. Параметр β следует трактовать, как интегральная оценка уровня «заражаемости» (сколько лиц может заразить один инфицированный) и уровня самоизоляции населения. Индекс Херста может принимать значения в диапазоне $0 < H < 1$. При этом в случае $H = 0.5$ процесс будет сопоставим с марковским. Общий поток, генерируемый в пограничном слое, который является входящим потоком в ядро, будет самоподобным [5, 7].

Исходящий поток пограничного слоя в терминологии СМО это поток ухода требований из системы. Учитывая, что уход требований можно идентифицировать как переход в ядро (на изоляцию в медучреждение), уход в связи с выздоровлением, уход по причине смерти и перемещения в другой узел, уместно также вводить несколько типов потоков. Аналогично описаниям процесса поступления требований в систему процесс ухода требований из системы может быть описан марковский случайный процесс.

Резюмируя, выше изложенное, СМО, располагающиеся в пограничном слое представляют собой системы, интерпретируемые согласно классификации по системе Кендала, как СМО типа M/Y/1. Один обслуживающий прибор трактуется как рассмотрение всего слоя как одна СМО.

2.3. Процессы распространения инфекции в регионе: ядро

Процесс поступления требований в ядро является самоподобным, однако самоподобие будет меньше выражено, чем в случае процесса заражения в пограничном слое. Это связано с тем, что не все инфицированные размещаются на изоляцию в медучреждение. Однако вероятность того, что если у кого-то из группы контактных лиц идентифицировали инфекцию и перевели в стационар, то в этой группе будут еще инфицированные, которые потребуют обсервации. Время обслуживания требований в СМО детерминированное (находится в некотором диапазоне). Нижняя граница – это минимальный срок, на который госпитализируют больного, верхняя – максимально возможный срок госпитализации (сопоставимым со временем излечения). Время «обслуживания» потока связано с производительностью узла. Интенсивность обслуживания зависит от количества «обслуживающих приборов». Количество «обслуживающих приборов» сопоставляется с количеством с койко-мест в медицинских учреждениях.

Превышение интенсивности входящего потока величины интенсивности потока обслуживания порождает образование очередей. В связи с тем, что специфика задачи не предполагает возможности ожидания, то это порождает потери в обслуживании. Потери связаны как с отсутствием свободных «приборов» для обслуживания, так и с истечением «времени жизни» некоторых требований. «Время жизни» – это временной интервал, в течение которого имеет смысл в обслуживании требования (в привязке к пандемии, смысл в госпитализации для критичных больных).

Исходящий поток ядра также является случайным процессом, т. к. является стационарным, ординарным и обладает свойством отсутствия последействия. При этом важно разделять поток на два типа: ухода требований из системы по причине «выздоравления» или «истечения времени жизни». Интенсивность последнего зависит от коэффициента загрузки обслуживаемых «приборов» СМО.

Дисциплина обслуживания требований в такой системе определяется как относительный приоритет, т. е. обслуживание приоритетного требования начинается только по завершению обслуживания требования уже находящегося в системе. Введение приоритетов потоков позволяет уточнить интенсивности потоков обслуживания, разделяя последние на необходимость медпомощи определенного уровня (что трактуется также в степени оснащенности койко-мест).

Резюмируя, выше сказанное, ядро системы в терминологии СМО может быть классифицировано по системе Кендала, как СМО типа Y/D/M/m. В зависимости от задач моделирования можно рассматривать ядро как совокупность всех медучреждений и одну СМО, а можно разделять на несколько СМО, каждая из которых будет ассоциироваться с отдельным медучреждением.

3. Процесс распространения инфекции как функционирование СМО

Процесс распространения вирусной инфекции в рамках сделанного выше моделирования следует рассматривать как функционирование двух последовательных СМО. В свою очередь, их функционирование связано со сменой состояний. Каждое из которых, определяется вероятностью наступления того или иного события в каждой СМО. Переход из одного состояния в другое осуществляется случайным образом и описывается с помощью матрицы вероятностей переходов. В зависимости от того, конечно или бесконечно число состояний, и сама матрица вероятностей либо конечна, либо бесконечна [2, 3]. Моделирование состояний в СМО позволяет: получить вероятности отдельных ее состояний, как в прогнозе, так и в текущий момент времени, оценить временные характеристики отдельных состояний системы, как например, время перехода из одного состояния в другое.

Наибольшую угрозу для общества (региона) представляет состояние системы, когда обслуживающие «приборы» ядра не справляются с потоком требований, поступающим из пограничного слоя (состояние перегрузки). Вероятность наступления этого состояния зависит от вероятности нахождения в смежном для него состоянии, когда в ядре обслуживаются требования и их количество предельно, относительно производительности ядра. Не допустить его позволяет либо увеличение производительности обслуживающих приборов, либо увеличение самих обслуживающих приборов, либо изменение характеристик потока, определяющего вероятность наступления.

Заключение

Данное исследование может стать основой для дальнейшего моделирования процессов распространения вирусной инфекции. Рассмотренный подход к моделированию, позволяет оценить вероятность наступления того или иного состояния системы. В свою очередь, это позволяет определить степень влияния тех или иных параметров потоков на вероятности состояния системы, рассчитать временные характеристики отдельных процессов. Точность моделирования зависит от частоты корректировки характеристик отдельных процессов.

Практическое значение подобного моделирования заключается в том, что оно позволит в дальнейшем контролировать процессы распространения инфекции. Это отражается в под-

боре параметров потоков и на их основе разработки комплекса мер реагирования, например, закрытие границ, повышение уровня самоизоляции населения, перевода на карантин пациентов на более ранней стадии заболевания. Более того, данный подход позволяет моделировать потоки распространения в зависимости от информации об обеспеченности медицинских учреждений и моделировать их потребности в зависимости от состояния системы.

Литература

1. *Авдеев, О. Н.* Моделирование систем : учеб. Пособие / О. Н. Авдеев, Л. В. Мотайленко. – Санкт-Петербург : Изд-во СПбГТУ, 2001. – 170 с.

2. *Азарнова, Т. В.* Оценка временных характеристик систем с сетевой топологией и стохастическими процессами функционирования / Т. В. Азарнова, В. В. Ухлова // Актуальные проблемы прикладной математики, информатики и механики : А43 сб. тр. Международной научной конф. (Воронеж, 17–19 декабря 2018 г.). – Воронеж, 2019. – С. 499– 509.

3. *Азарнова, Т. В.* Алгоритм оценки времени задержки передачи трафика в телекоммуникационных сетях на основе стохастического моделирования процесса обслуживания / Т. В. Азарнова, В. В. Ухлова, А. Л. Ухин // Системы управления и информационные технологии. – 2015. – № 59(1). – С. 44–49.

4. *Бусленко, И. В.* Лекции по теории сложных систем / И. В. Бусленко, В. В. Калашников, И. Н. Коваленко. – Москва : Сов. Радио. 1973. – 439 с.

5. *Калашников, В. В.* Математические методы построения стохастических моделей обслуживания / В. В. Калашников, Т. Рачев. – Москва : Наука, 1988. – 311 с.

6. *Корякин, И.* Коронавирус вписался в математическую модель. – Режим доступа: https://www.kommersant.ru/doc/4277027. – (Дата обращения: 10.03.2020).

7. Статья Википедии о вирусной инфекции COVID-19. – Режим доступа: https://ru.wikipedia.org/wiki/COVID-19. – (Дата обращения: 10.04.2020).

8. *Ухлова, В. В.* Математические модели и методы оптимизации временных характеристик сложных стохастических систем с сетевой топологией : диссертация ... кандидата физико-математических наук : 05.13.18 / Ухлова В. В. – Елец, 2011. – 189 с.

Палкина София Александровна – студентка 1-го курса факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: mail.soffya@mail.ru

Ухлова Вера Владимировна (научный руководитель) – канд. физ.-мат. наук, доц. кафедры математических методов исследования операций Воронежского государственного университета. E-mail: it_content@list.ru

УДК 004.932

АЛГОРИТМ КОРРЕКЦИИ ПЕРСПЕКТИВЫ НА ИЗОБРАЖЕНИИ

Т. С. Подакина

Воронежский государственный университет

В статье рассмотрена одна из актуальных проблем цифровой обработки изображений – исследование и улучшение алгоритма коррекции перспективы. Для решения поставленной задачи были исследованы виды перспектив, существующие алгоритмы коррекции и разработана программа для получения нового, видоизмененного изображения.

Введение

Перспектива – это способ изображения трёхмерных предметов и объектов в двухмерном пространстве с учётом принципов и законов зрительского восприятия человеком окружающего пространства. Такой способ построения изображения базируется на расположении элементов изображения относительно линии горизонта и лежащей на ней точки зрения. В соответствии с законами линейной перспективы, линии параллельные друг другу, но не параллельные линии горизонта сходятся в пространстве в так называемой точке схода. Это позволяет изображать объекты и предметы в плоскости листа (холста) так, как они воспринимаются человеческим глазом в природе. Таким образом, удаляющиеся в глубину от зрителя предметы и их части выглядят на картине меньше, по сравнению с аналогичными предметами, расположенными на первом плане.

Разделяют несколько видов перспективы.

1) Прямая линейная перспектива. Это традиционный вид перспективы, который является привычным для человека.

2) Панорамная перспектива. Изображение, строящееся на внутренней цилиндрической (иногда шаровой) поверхности.

3) Сферическая перспектива. Особый способ организации изобразительного пространства на плоскости картины или росписи стены, свода, купола, который заключается в иллюзорном углублении зрительного центра, совпадающего с геометрическим центром композиции, и расположении остальных элементов в воображаемом сферическом пространстве.

На цифровом изображении эффект перспективы появляется за счёт оптических свойств объектива.

Рассмотрим рис. 1, на котором из-за неверного ракурса и в следствие перспективы, нижняя часть дома кажется намного больше верхней части, а сам дом визуально имеет наклон. Для коррекции таких искажений на изображении используют коррекцию перспективы. Главной задачей является растяжение верхней части изображения и соразмерное сжатие нижней части. При этом воздействия на другие объекты в сцене, не требующие коррекции, воздействия не должно быть оказано.

Анализ и коррекция перспективы

Для коррекции перспективы необходимо осуществить сдвиг проблемной части изображения в необходимую сторону. Определение таких проблемных частей изображения может быть

Рис. 1. Неправильная перспектива изображения

осуществлено программно или с помощью человека. Используя попиксельную обработку изображения, можно избежать больших погрешностей в вычислениях и произвести растяжение/сжатие проблемных областей изображения. Рассмотрим алгоритм коррекции изображения, использующий попиксельную обработку изображения. На рис. 2 изображена шахматная доска. Задачей алгоритма является сдвиг центрального пикселя на некоторое расстояние вправо.

Началом алгоритма является разделение исходного изображения на четыре части, используя входной пиксель изображения, который определяет проблемную часть исходного изображения (рис. 3).

Дальнейшая работа будет производиться с каждой из частей изображения отдельно. После обработки каждой части происходит их соединение для получения цельного изображения (рис. 4).

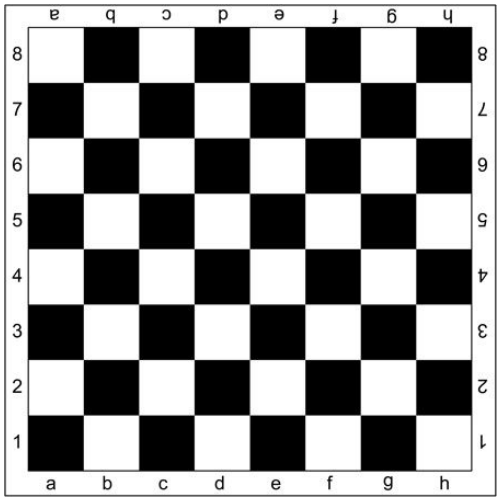


Рис. 2. Шахматная доска

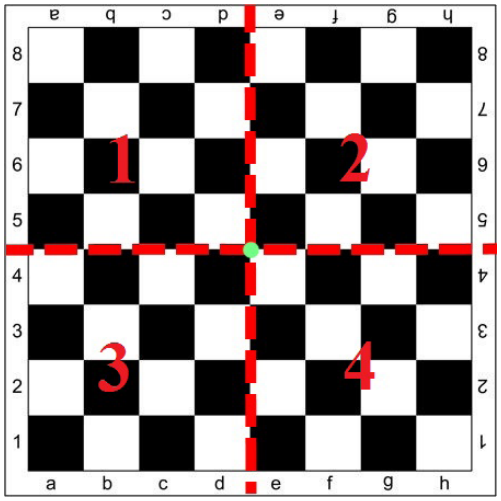


Рис. 3. Определение пикселя и разбиение изображения на четыре части

В зависимости от сдвига пикселя относительно исходного положения, определяется одна из двух вариаций алгоритма, сжатие или растяжение, рассмотренные ниже.

Вне зависимости от вариации алгоритма, для дальнейшей работы необходимо вычислить коэффициент сдвига f, который рассчитывается по формуле:

$$f = \frac{s}{h},$$

где s – количество пикселей, на которое происходит сдвиг исходного пикселя, а h – высота исходного изображения.

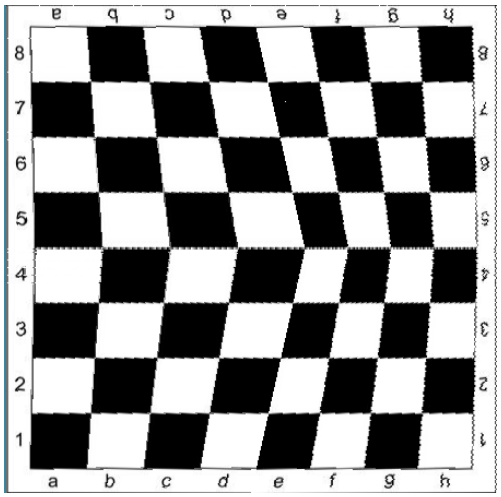


Рис. 4. Ожидаемый результат работы алгоритма коррекции

1) Растяжение нижней части изображения (данная вариация алгоритма подходит и для случая, когда необходимо сжать верхнюю часть изображения, таким образом применяется для 1 и 4 частей изображения) (рис. 3):
Для такого типа растяжения обрабатывая каждую строку алгоритма необходимо осуществлять изменение ширины изображения с помощью формулы:

$$rt = rt + f,$$

где rt – количество пикселей, на которое увеличивается ширина исходного изображения с каждым шагом, начально значение равно нулю.

Таким образом каждая строка изображения имеет пропорционально увеличивающуюся ширину, что приводит к равномерному растяжению.

2) Сжатие нижней части изображения (данная вариация алгоритма подходит для случая, когда необходимо сжать верхнюю часть изображения, таки образом применяется для 2 и 3 частей изображения) (рис. 3):
Для сжатия нижней части изображения необходимо пропорционально уменьшать ширину обработанного изображения, используя формулу:

$$wt = nw - rt,$$

где wt – ширина изображения на каждом шаге, nw – максимальная ширина изображения, полученная путем сложения расположения входного пикселя с числом пикселей для сдвига.

Для масштабирования изображения на каждом шаге можно использовать один из способов интерполяции [1]:

– линейная интерполяция;

– кубическая интерполяция;

– билинейная интерполяция.

Рассмотрим один из самых часто используемых алгоритмов масштабирования в компьютерной графике: билинейная интерполяция. Ее достоинством является компенсация ошибок перемещения по координатам. Для использования такой интреполяции необходимо знать значения четырех окрестных пикселей.

На каждом шаге создания нового пикселя обработанного изображения определяем соответствующий ему пиксель исходного изображения по формуле:

$$tmpW = \frac{i}{(nw-1)*(ow-1)},$$

$$tmpH = \frac{i}{(nh-1)*(oh-1)},$$

где ow – ширина исходного изображения, oh – высота исходного изображения, nh – высота нового изображения, (i, j) – координаты текущего пикселя нового изображения.

Для вычисления массива коэффициентов, которые будут применяться для определения значений цвета нового пикселя используются формулы:

$$k1 = (1 - tmpW) * (1 - tmpH),$$
$$k2 = tmpW * (1 - tmpH),$$
$$k3 = tmpW * tmpH,$$
$$k4 = (1 - tmpW) * tmpH.$$

После получения коэффициентов необходимо вычислить новое значение пикселя, основываясь на значении пикселя исходного изображения, а также его соседей. Для этого необходимо вычислить значения красного, зеленого и синего цветов в пикселе. Эти значения являются представлением RGB – модели изображения, которая является аддитивной цветовой моделью, описывающей способ кодирования цвета [2]. После получения цветов необходимых пикселей, используем коэффициенты, полученные выше, для вычисления новых значений составляющих:

$$r = r1 * k1 + r2 * k2 + r3 * k3 + r4 * k4,$$
$$g = g1 * k1 + g2 * k2 + g3 * k3 + g4 * k4,$$
$$b = b1 * k1 + b2 * k2 + b3 * k3 + b4 * k4,$$

где (r, g, b) – новые значения компонент цветов пикселя, $(r1, g1, b1)$ – значения красного, зеленого и синего цветов в пикселе исходного изображения, $(r2, g2, b2)$ – соседнего справа пикселя к исходному, $(r3, g3, b3)$ – пикселя, находящегося на один шаг вправо и один шаг вниз от исходного, $(r4, g4, b4)$ – соседнего снизу пикселя.

Таким образом, создавая новое изображение с корректированной перспективой необходимо построчно изменять исходное изображение. На рис. 5 изображен пример работы вышеописанного алгоритма, реализованного на языке Java. Части 1 и 4 имеют растяжение нижней части, части 2 и 3 сжатие нижней части.

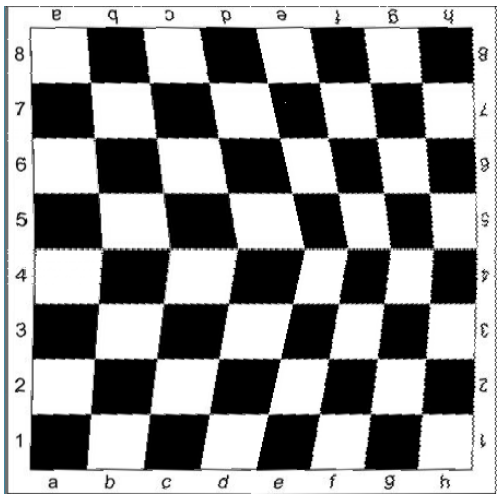


Рис. 5. Результат работы реализованного алгоритма

Заключение

В заключении хотелось бы отметить, что существующий алгоритм может применяться к любому виду перспективы изображений, что является его преимуществом относительно других алгоритмов. Из недостатков работы алгоритма можно выделить потерю качества обработанного изображения в связи с приведением к численному типу коэффициентов для расчёта новых показателей ширины.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Половко, А. М.* Интерполяция. Методы и компьютерные технологии их реализации / А. М. Половко, П. Н.Бутусов. – Санкт-Петербург: БВХ, 2004. – 320 с.

2. *Мураховский В. И.* Большая книга цифровой фотографии / В. И. Мураховский С. В. Симонович. – Санкт-Петербург: Питер, 2011. – 303 с.

Подакина Татьяна Сергеевна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: podakina.t@mail.ru.

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: mitp@amm.vsu.ru.

УДК 519.854.2

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЭВРИСТИЧЕСКИХ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА

А. В. Полозова

Воронежский государственный университет

Введение

Данная статья несет за собой цель в виде ознакомления и анализа жадного и муравьиного в контексте решения задачи коммивояжера (ЗК), а также представления первичных результатов тестирования данных алгоритмов и их сравнения между собой. В данной работе представлена постановка рассматриваемой задачи, описание алгоритмов решения и вычислительный эксперимент, поставленный на большом количестве входных данных.

1. Задача коммивояжера

Рассматривается задача, в которой курьер обязан привезти товар в конкретное количество пунктов доставки и возвратиться в место отправления. Требуется установить, в каком порядке он обязан обойти пункты доставки, для того чтобы его путь был минимальной длины.

Формализацию задачи коммивояжера как задачи математического программирования можно представить следующим образом [2].

Пусть дано:

$I = \{1,...,N\}$ – множество пунктов доставки;

$C = \{C_{ij} : i, j = \overline{1,N}, i \neq j\}$ – матрица стоимостей перехода из города i в город j.

Переменные задачи:

$$x_{ij} = \begin{cases} 1, & \text{если из пункта } i \text{ имеет место перемещение в пункт } j, \\ 0, & \text{иначе.} \end{cases}$$

$$L = \sum_{i \in I} \sum_{j \in I} c_{ij} x_{ij} \to \min \tag{1}$$

при ограничениях:

– только *один подъезд* к пункту доставки

$$\sum_{i \in I} x_{ij} = 1, \forall j \in I, \tag{2}$$

– только *один выезд* из пункта доставки

$$\sum_{j \in I} x_{ij} = 1, \forall i \in I. \tag{3}$$

В целях устранения появления частичных замкнутых маршрутов вводятся дополнительные переменные задачи:

$u_i \geq 0, u_i \in \mathbb{N}$ – номер шага, на котором посетили город i.

Тогда исключение подциклов полиномиальным числом ограничений может быть представлено в виде:

$$u_i - u_j + n x_{ij} \leq n - 1, i, j \in I \setminus i_1, i \neq j. \tag{4}$$

2. Обзор жадного алгоритма

Жадный алгоритм – алгоритм, заключающийся в принятии локально оптимальных решений на каждом этапе, допуская, что конечное решение также окажется оптимальным [1].

Один из вариантов жадного алгоритма в задаче коммивояжера при выборе очередного города берёт ближайший не посещённый до этого город.

Рассмотрим для примера сеть на рис. 1, представляющую узкий ромб. Пусть Коммивояжер стартует из города 1. Алгоритм «иди в ближайший город» выведет его в город 2, т. к. длина пути 3 меньше, чем 5, затем в город 3, затем в 4; на последнем шаге придется платить за жадность, возвращаясь по длинной диагонали ромба. В результате получится не кратчайший путь 1-2-4-3-1 длиной 11, а более длинный путь 1-2-3-4-1 длиной 15.

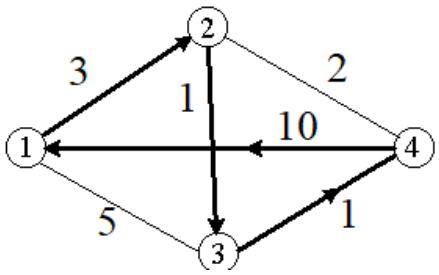


Рис. 1. Схематическое представление сети городов

3. Обзор муравьиного алгоритма

Рассмотрим принцип действия муравьиного алгоритма при решении задачи коммивояжёра [3].

Пусть муравей находится в узле i, а узел j – это один из узлов, доступных для перехода. Стоимость перехода из узла i в узел j обозначим C_{ij}. А интенсивность феромона на пути из i в j обозначим как τ_{ij}. Исходя из того, что кратчайший путь не должен содержать циклов, введём множество S_i, которое назовём множеством вершин, допустимых для перехода из вершины i.

Тогда вероятность перехода муравья из i в j будет равна:

$$p_{ij} = \frac{\tau_{ij}^{\alpha} + \frac{1}{c_{ij}^{\beta}}}{\sum_{l \in S} \left(\tau_{il}^{\alpha} + \frac{1}{c_{il}^{\beta}} \right)}, \tag{5}$$

где α и β – это регулируемые параметры, определяющие важность составляющих (стоимости перехода и уровня феромонов) при выборе пути. Выбор правильного соотношения параметров является предметом исследований, и в общем случае производится на основании опыта.

После того, как муравей успешно проходит маршрут, он оставляет на всех пройденных ребрах след, обратно пропорциональный длине пройденного пути:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k}, & \text{для } (i,j) \in T_k(t), \\ 0, & \text{для } (i,j) \notin T_k(t), \end{cases} \tag{6}$$

где L_k – длина пути k-го муравья, Q – регулируемый параметр, а $T_k(t)$ – маршрут, пройденный муравьём k на итерации t, $T_k(t)$ – длина этого маршрута.

Поиск кратчайшего пути с использованием формулы (5) позволяет избежать выбора на каждом шаге вершины с наилучшей видимостью (расстояние до которой является наименьшим из возможных для данной вершины) за счёт использования опыта предыдущих поколений, который отражен в уровне феромонов $\tau_{ij}(t_k)$.

Кроме этого, следы феромона испаряются, то есть интенсивность феромона на всех ребрах уменьшается на каждой итерации алгоритма. Таким образом, в конце каждой итерации необходимо обновить значения интенсивностей. Количество феромона на ребре $(i,\ j)$ на итерации t обозначается как $\tau_{ij}(t)$.

Правило обновления феромона выглядит следующим образом:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}^{k}(t) + \rho * \Delta\tau_{ij}^{k}, \tag{7}$$

где ρ – параметр, характеризующий испарение, $\rho \in [0;1]$.

4. Вычислительный эксперимент

Для оценки работы вышеописанных алгоритмов в данной работе было проведено два типа тестирования: интенсивное тестирование и тестирование на данных с уже известным точным решением [4].

1. Интенсивное тестирование – тестирование на больших объемах тестовых данных с целью сравнения работы алгоритмов между собой.

Для проведения данного типа тестирования была разработана программа для генерирования 100 симметричных матриц с нулевой диагональю, заполненных случайными числами от 1 до N, где N – это размерность матрицы. В нашем случае сгенерированные матрицы представляют собой матрицы расстояний между городами (пунктами доставки). Полученные матрицы записываются в текстовый файл, который в последствии является входными данными для основной программы с реализацией алгоритмов решения.

Для тестирования было сгенерировано 5 файлов, содержащих матрицы размерностью 5, 10, 25, 50, 100. В процессе тестирования находится решение для каждой из 100 матриц в виде пути и его длины, а также вычисляется среднее значение длин найденных решений. Алгоритмы запускаются при разных наборах параметров, таких как: количество городов (размерность матриц), количество муравьиных колоний, максимальное число итераций. Именно этими данными мы будем оперировать при сравнении алгоритмов. Результаты тестирования приведены в табл. 1.

2. Второй тип тестирования заключается в том, чтобы протестировать работу алгоритмов и сравнить результаты с уже известными точными решениями. Данное тестирование позволит определить, какой алгоритм дает результат, наиболее приближенный к точному решению. Результаты тестирования приведены в табл. 2.

В табл. 1 представлены результаты тестирования муравьиного и жадного алгоритмов с несколькими настраиваемыми параметрами. Алгоритмы были запущены для разного количества городов, для 10 и для 100 муравьиных колоний и с указанием максимального количества итераций в размере 10 и 100. Можно увидеть, что муравьиный алгоритм дает лучшее решение, то есть меньшее значение длины пути, чем жадный алгоритм даже при минимальном количестве муравьёв и итераций. Также можно проанализировать влияние количества колоний на результат муравьиного алгоритма, сравнив, например, эксперимент № 2 (10 муравьёв) и эксперимент № 7 (100 муравьёв), где видно, что путь, полученный в рамках эксперимента № 7 имеет меньшую длину (49,3 против 54,2).

Такая зависимость результата от числа муравьёв перетекла из живой природы в математическую модель. Метод муравьиных колоний, применяемый для решения задач оптимизации,

Таблица 1

Результаты интенсивного тестирования муравьиного и жадного алгоритмов

№ эксперимента	Кол-во городов	Кол-во муравьёв	Макс кол-во итераций	Муравьиный алгоритм	Жадный алгоритм
1	5	10	10	14	17,5
2	10	10	10	54,2	67,1
3	25	10	10	202,2	320,5
4	50	10	10	805,5	1304,3
5	100	10	10	3189,1	5966,1
6	5	100	10	14	17,5
7	10	100	10	49,3	67,1
8	25	100	10	193,7	320,5
9	50	100	10	737	1304,3
10	100	100	10	2470,2	5966,1
11	5	10	100	14	17,5
12	10	10	100	52,1	67,1
13	25	10	100	198,9	320,5
14	50	10	100	753,8	1304,3
15	100	10	100	2338,1	5966,1
16	5	100	100	14	17,5
17	10	100	100	46,8	67,1
18	25	100	100	168,3	320,5
19	50	100	100	712,7	1304,3
20	100	100	100	2005,4	5966,1

Таблица 2

Результаты тестирования муравьиного и жадного алгоритмов на данных с известным точным решением

№ эксперимента	Кол-во городов	Кол-во муравьёв	Муравьиный алгоритм	Жадный алгоритм	Точное решение
1	5	10	20	25	19
2	15	10	359	639	291
3	17	10	2390	11887	2085
4	26	10	964	4327	937
5	42	10	882	4973	699
6	48	10	35443	332442	33523
7	5	100	20	25	19
8	15	100	332	639	291
9	17	100	2181	11887	2085
10	26	100	956	4327	937
11	42	100	803	4973	699
12	48	100	34540	332442	33523

основан на идеях, многие из которых в живой природе осуществляет популяция муравьёв. И большее количество особей, очевидно, с большей вероятностью доберется до цели за меньшее количество времени, чем меньшее количество особей.

Следующим изменяемым параметром является заданное максимальное количество итераций. Увеличивая это число, можно добиться получения все лучшего и лучшего результата, однако, это не всегда является оправданным. При увеличении числа итераций, увеличивается и время работы алгоритма, при этом увеличение количества итераций от 100 и более, влечет за собой медленное, почти незначительное сокращение длины пути, а временные и производственные потери могут быть значительными. В результате тестирования было выявлено оптимальное количество итераций, и оно составляет приблизительно 100 итераций для количества городов от 10 до 100.

Заключение

По результатам тестирования на данных, где было известно точное решение, можно с уверенностью сказать, что муравьиный алгоритм дает заметно лучшее решение в сравнении с жадным алгоритмом. То же самое можно сказать и про результаты интенсивного тестирования. Здесь также можно заметить, что увеличение количества муравьёв и итераций позволяет муравьиному алгоритму «улучшить» полученное решение.

Однако при небольшом количестве городов различия между результатами не слишком существенные, это можно увидеть при обоих типах тестирования. И, учитывая преимущество жадного алгоритма в легкости реализации и во времени работы, можно отметить, что использование жадного алгоритма для некоторых задач с небольшой размерностью может оказаться предпочтительнее.

Работа выполнена под руководством кандидата физико-математических наук Медведевой О. А.

Литература

1. Алгоритмы: построение и анализ = Introduction to Algorithms / Кармен Т. [и др.]; отв. ред. И. В. Красиков.– 2-е изд – М.: Вильямс, 2005. – 1296 с.

2. *Корбут А. А., Финкельштейн Ю. Ю.* Дискретное программирование. – М. : Наука, 1969. – С. 258–264.

3. *Пантелеев, А. В.* Разработка алгоритмического и программного обеспечения метода муравьиных колоний / А. В. Пантелеев, Е. А. Алёшина // Научный вестник МГТУ ГА. – 2008. – С. 40–42.

4. TSP Data for the Traveling Salesperson Problem. – Режим доступа: https://people.sc.fsu.edu/~jburkardt/datasets/tsp/tsp.html. (Дата обращения: 16.02.2020).

Полозова Анастасия Викторовна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: polozova.av@gmail.com

Медведева Ольга Александровна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 616

АНАЛИЗ МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ КЛАССИФИКАЦИИ РАСПРОСТРАНЕННЫХ БОЛЕЗНЕЙ ГРУДНОЙ КЛЕТКИ

В. В. Поляков

Воронежский государственный университет

Введение

Целью работы является создание модели на основе алгоритмов глубокого обучения) и непосредственная тренировка ее на имеющихся наборах данных для получения наилучших результатов для походящих данной задаче метрик.

При построении моделей использовались архитектуры сверточных нейронных сетей и модели с передачей обучения

Метрики подобраны с учетом имеющихся данных.

1. Машинное обучение и глубокое обучение

1.1. Машинное обучение

Машинное обучение – это научная область, изучающий методы построения алгоритмов, способных обучаться на основе имеющихся данных. Говорят, что компьютерная программа обучается на основе опыта E по отношению к некоторому классу задач T и некоторой мере эффективности P, если эффективность программы при решении задач из T, измеряемая с помощью P, повышается с опытом E (определение Машинного Обучения, Том Митчел, 1997).

Существует два основных класса алгоритмов машинного обучения:

- Обучение с учителем.
- Обучение без учителя.

Обучение с учителем является одним из наиболее часто используемых методов машинного обучения, оно используется, когда необходимо предсказать определенный результат по данному объекту. Модель машинного обучения строится на основе пар объект-ответ, которые представляют обучающий набор данных. Цели обучения – получение точных прогнозов на ранее не встречавшихся данных. Две основные задачи машинного обучения с учителем – классификация (прогнозирование принадлежности объекта к определенному классу из списка возможных вариантов) и регрессия (прогнозирование непрерывного числа или числа с плавающей точкой).

Обучение без учителя включает в себя все виды машинного обучения, когда ответ неизвестен, в машинном обучении есть лишь входные данные и алгоритму необходимо извлечь знания из этих данных.

В данном случае рассматривается задача классификации.

1.2. Глубокое обучение

Глубокое обучение – совокупность широкого семейства методов машинного обучения, основанных на имитации работы человеческого мозга в процессе обработки данных и создания паттернов, используемых для принятия решений. На больших данных глубокое обучение по-

казывает более высокую точность результатов в сравнении с традиционными алгоритмами машинного обучения.

Данный класс алгоритмов машинного обучения использует многослойную систему нелинейных фильтров, для извлечения признаков с преобразованиями, каждый последующий слой получает входные данные предыдущего слоя.

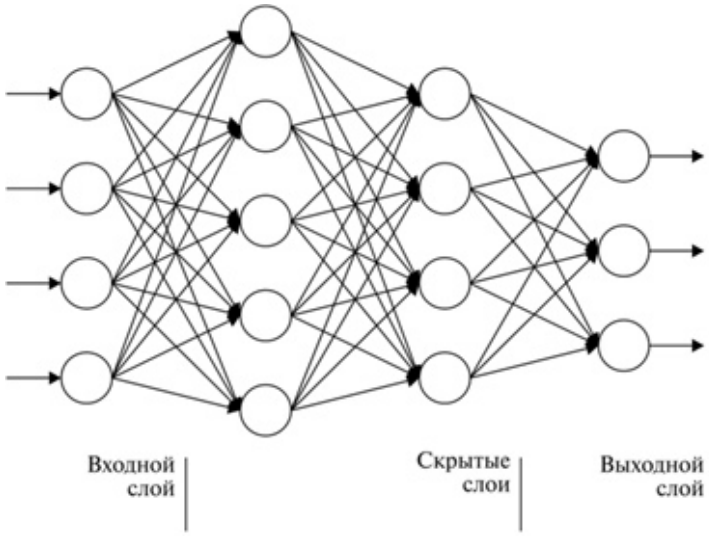


Рис. 1. Пример глубокой нейронной сети

Слои моделей формируются в процессе обучения для выявления признаков на нескольких уровнях представлений, которые соответствуют различным уровням абстракции; при этом признаки организованы иерархически – признаки более высокого уровня являются производными от признаков более низкого уровня.

2. Архитектура сверточных нейронных сетей

2.1. Свертка

Сверточная нейронная сеть – специальная архитектура нейронных сетей, изначально нацеленная на эффективное распознавание изображений.

Свертка – операция над парой матриц A (размера $n_x n_y$) и B (размера $m_x m_y$), результатом которой является матрица $C = A * B$, размера $(n_x - m_x + 1) \times (n_y - m_y + 1)$. Каждый элемент вычисляется как скалярное произведение матрицы B и некоторой подматрицы A такого же размера. То есть $C_{i,j} = \sum_{u=0}^{m_x-1} \sum_{v=0}^{m_y-1} A_{i+u,j+v} B_{u,v}$. Матрицу A называют изображением, а B фильтром или ядром.

По сути фильтр перемещается вдоль изображения и определяет, присутствует ли некая искомая характеристика в конкретной его части.

2.2. Сверточный слой

Сверточный слой нейронной сети представляет из себя применение операции свертки к выходам предыдущего слоя, где веса ядер являются обучаемыми параметрами.

Можно заметить, что применение операции свертки уменьшает изображение. Также пиксели, находящиеся на границе изображений, учувствуют в меньшем количестве сверток, чем внутренние. В связи с этим в сверхточных слоях нейронной сети может использоваться дополнение изображений. Изображения, являющиеся выходами предыдущих слоев, дополняются н

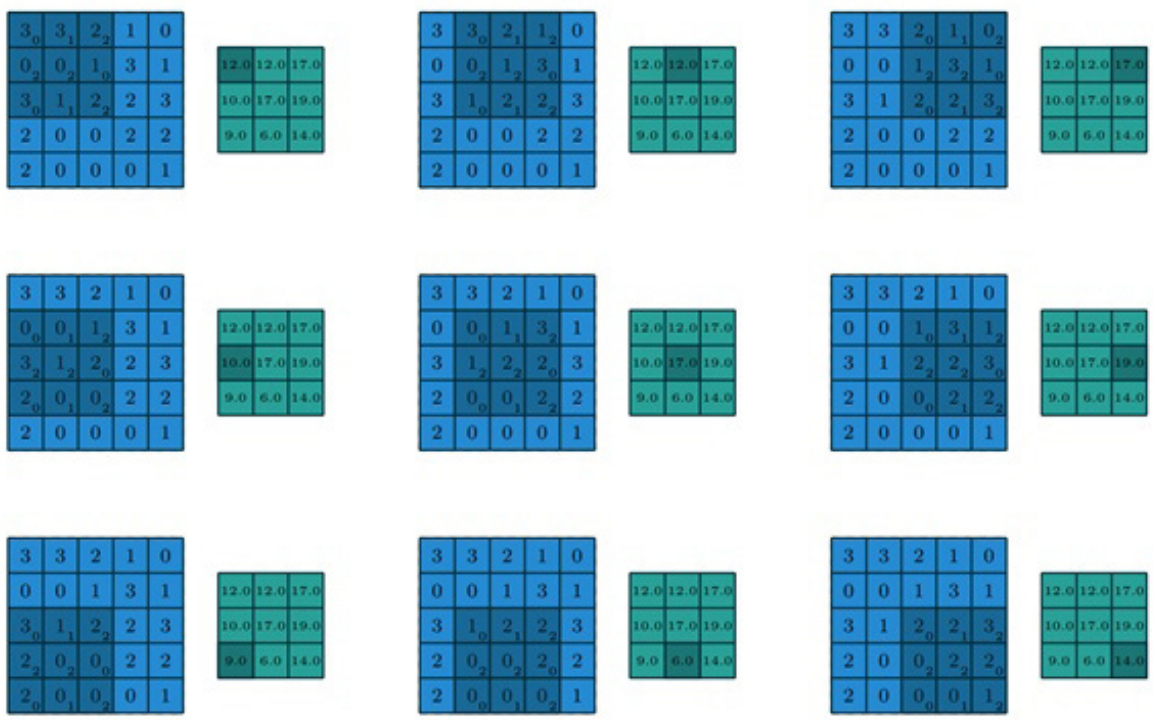


Рис. 2. Пример свертки матрицы 5×5 *матрицей размера* 3×3

пикселями так, чтобы при свертке сохранялся размер изображения. Такие свертки называются одинаковыми, свертки без дополнения называются правильными.

Если на вход сверточному слою поступает изображение, состоящие из нескольких каналов, фильтры применяются к каждому из них. Выходом сверточного слоя является изображение, с количеством каналов, соответствующим количеству примененных входным данным фильтров. При создании данного слоя, указывается количество фильтров и их размеры.

2.2. Пулинговый слой

Пулинговый слой призван снижать размерность изображений, исходное изображение делится на блоки размером $w\times h$ и для каждого блока вычисляется некоторая функция. Чаще всего используются функции максимума и среднего. У этого слоя нет обучаемых параметров. Основные цели пулингового слоя:

- Уменьшение изображения, чтобы последующие свертки оперировали над большей областью исходного изображения.
- Увеличение инвариантности выхода сети по отношению к малому переносу входа.
- Ускорение вычислений.

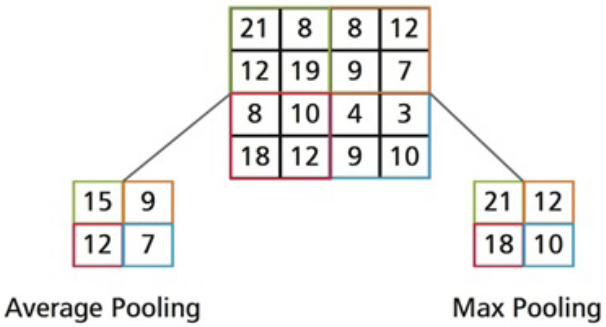


Рис. 3. Примеры пулинга

3. Работа с данными

3.1. NIH Chest X-ray Dataset

Этот набор данных представляет собой 112120 изображений, более чем 30000 различных пациентов. Данные могут принадлежать к 15 классам, при этом 14 из них – классы болезней. Одно изображение может одновременно иметь несколько меток классов. Основная проблема этого набора изображений – его несбалансированность. Так из 112120 изображений 51759 изображений не имеют болезней. Если отдельно выделить случаи, когда одно изображение имеет лишь одну метку класса болезни, можно увидеть, что наиболее объемный класс болезни (инфильтрация легких) имеет около 9500 изображений, а наименее объемный (грыжа) – менее тысячи.

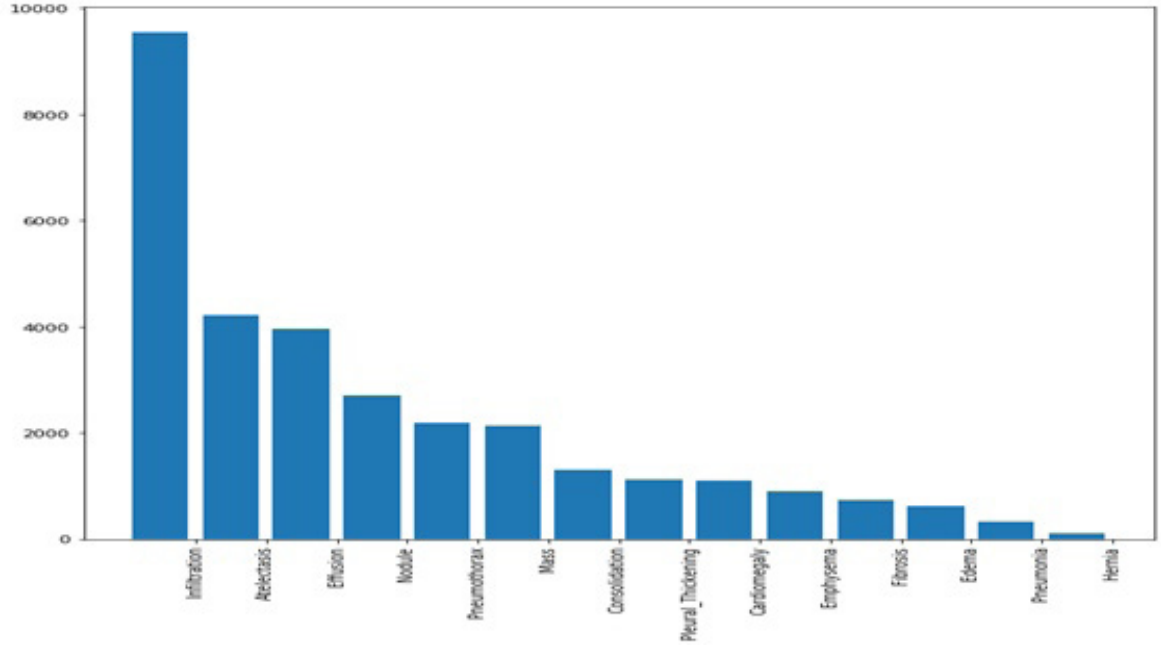


Рис. 4. Гистограмма частоты болезней

Также, как завил из работников медицинской сферы, далеко не все метки классов расставлены верно, что осложняет обучение нейронной сети.

3.2. Chest X-ray Images (Pneumonia)

Данные содержат 5863 изображения, которые могут иметь всего 2 метки классов – наличие или отсутствие пневмонии.

Набор также не сбалансирован, 1583 изображений не имеют пневмонии. Также, проблемой при тренировке нейронной сети является малочисленность данного датасета.

4. Модели

4.1. Модель для бинарной классификации первого датасета

Первый выбранный способ создания модели сверточной нейронной сети – вместо классификации каждой болезни попытка научать нейронную сеть находить аномалии в изображениях, то есть тренировка для ответа на вопрос – болеет ли человек на изображении или нет. К сожалению данная постановка задачи является слишком сложной для нейронной сети. Сверточная нейронная сеть находит паттерны и на их основе предсказывает какой-либо класс, однако в этом случае

сеть должна обратить внимание на слишком много несвязанных между собой частей изображения с разными признаками болезней, в связи с чем не может правильно обучиться.

Как метрики оценки правильности модели было решено использовать accuracy (сильного дисбаланса в отношении больных и здоровых в изображениях нет) и roc-кривую (стандартные метрики для оценки отношения правильно определенных классов).

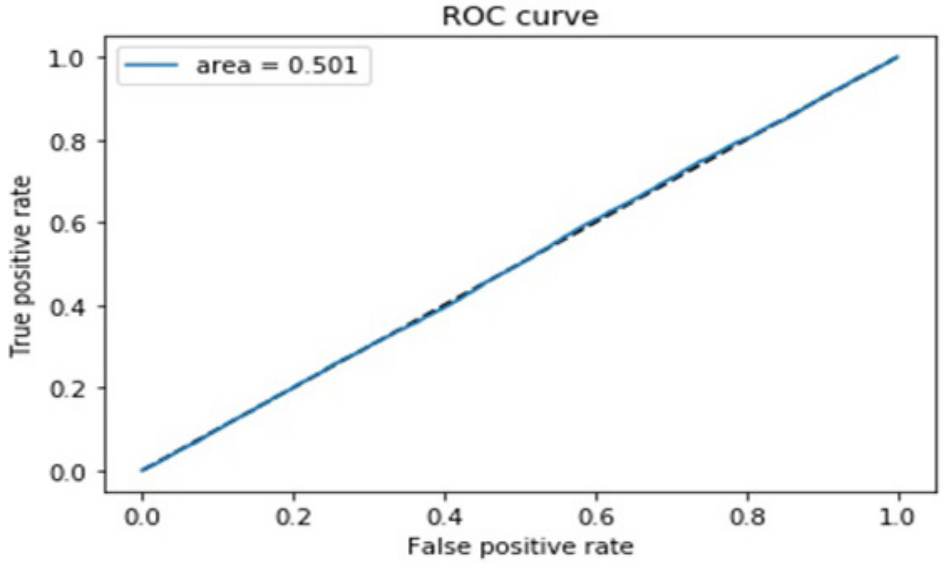


Рис. 5. Кривая ошибок для случая бинарной классификации первого датасета

Вычисленное значение accuracy – 60 %, следовательно, нейронная сеть близка к случайному определению маркеров классов.

4.1. Модель для мульти-классовой классификации первого датасета

В случае второго подхода к созданию модели светочной нейронной сети, произведена попытка решить задачу мульти-классовой классификации. Для начала, уберем из набора данных изображения, помеченные как отсутствие болезней, будем полгать, что, если при предсказании не было найдено ни одного класса ни одного класса из предложенных – пациент здоров. При обучении, нейронная сеть показывает неплохие результаты метрики accuracy, как на тренировочном, так и на валидационном наборе (около 88 %). Однако, в отличии от прошлой модели сверточной нейронной сети, данная метрика не может показать правильность нейронной сети. Как и говорилось ранее, данный набор сильно не сбалансирован, это означает, что высокие значения accuracy можно получить, предсказывая мажоритарные классы. Данное заключение подтверждается выводом кривой ошибок.

4.3. Модель бинарной классификации второго датасета

Для данной задачи нахождения пневмонии, была использована сверточная нейронная сеть архитектуры VGG16, которая была предобучена на наборе данных ImageNet, состоящего из более чем 14 миллионов изображений, разбитых на более 20000 тысяч классов. Первые сверточные слои такой модели уже умеют распознавать простые линии на изображении, следовательно, модели необходимо лишь натренировать более глубокие слои. Однако недостаток данных вместе с их несбалансированностью даже при учете обученной заранее модели, приводит к неточным результатам предсказаний. Так accuracy в данном случае равна 87 %, когда как кривая ошибок говорит об неспособности модели сделать верный прогноз.

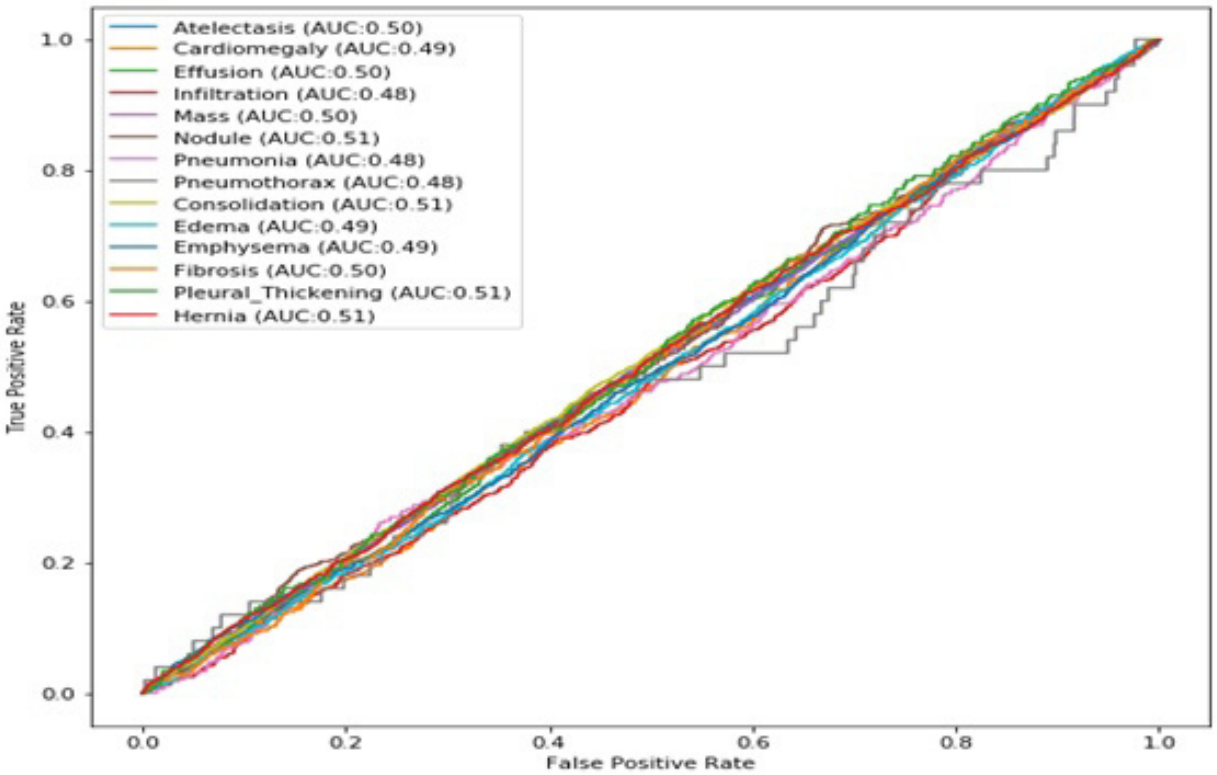


Рис. 6. Кривая ошибок для случая мульти-классовой классификации первого датасета

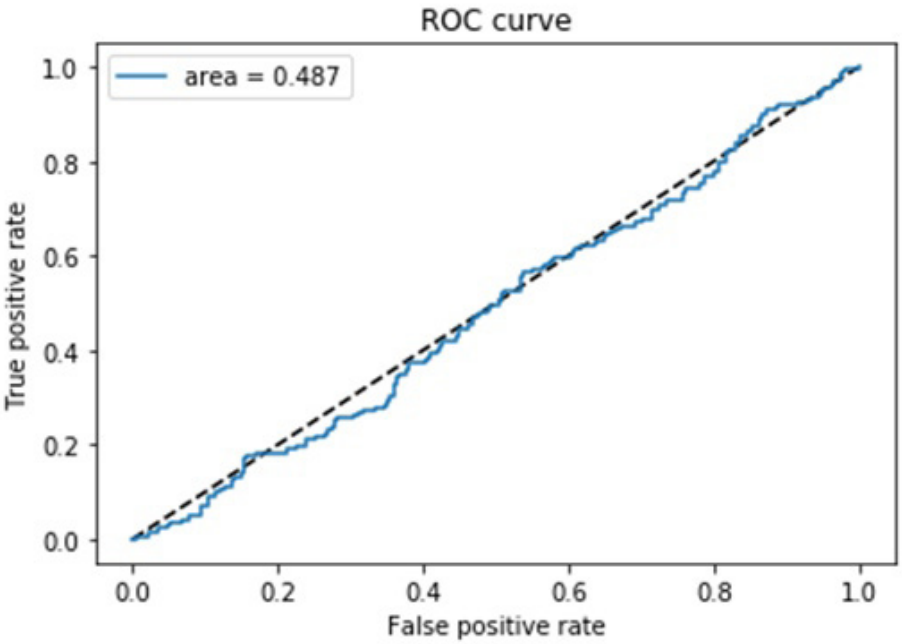


Рис. 7. Кривая ошибок для случая бинарной классификации второго датасета

Заключение

Несмотря на то, что свертка является мощным инструментом при создании нейронной сети для классификации изображений, ее моделям сложно обучаться на плохо подготовленных данных. К сожалению, на данный момент в открытом доступе нет набора данных, на котором можно было бы эффективно обучить светочную нейронную сеть.

Литература

1. *Мюллер, А.* Введение в машинное обучение с помощью Python / А. Мюллер. С. Гвидо. – издание 2017 года, Москва. – 393 с.

2. Сайт контроля версий. – Режим доступа: – https://github.com

3. Платформа для просмотра курсов онлайн. – Режим доступа: – https://www.coursera.org

4. Документация фреймворка керас. – Режим доступа: – https://keras.io

5. Документация фреймворка тензерфлоу. Режим доступа – https://www.tensorflow.org

6. Платформа для даса саентистов кэггл. Режим доступа – https://www.kaggle.com

7. Сайт для публикации статей о дата саенс. Режим доступа – https://towardsdatascience.com

Поляков Вадим Витальевич – студент 3-го курса кафедры Математического обеспечения ЭВМ Воронежского государственного университета. E-mail: vadpolyoab@gmail.com

Светлана Юрьевна Болотова (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры Математического обеспечения ЭВМ Воронежского государственного университета. E-mail: bolotova.svetlana@gmail.com

УДК 004.75

ПРИМЕНЕНИЕ ОБЛАЧНЫХ ТЕХНОЛОГИИ ПРИ ПОИСКЕ ДАННЫХ ПО РАЗЛИЧНЫМ КРИТЕРИЯМ

А. В. Попова, Е. В. Трофименко

Воронежский государственный университет

Введение

В настоящее время объем данных, который требуется хранить, а также обрабатывать, растет с невероятной скоростью по сравнению с производительностью компьютеров, которые эту информацию обрабатывают, поэтому возникают новые технологии для обработки этих данных, помогающие при минимальных системных требованиях обрабатывать большое количество данных с большой скоростью. Одной из таких технологий является Apache Spark.

Благодаря наличию различных инструментов для аналитической обработки данных, Apache Spark позволяет составлять различные отчеты в бизнес-приложениях, например, количество пользователей, просматривающих товары, тем самым можно составить отчеты по популярности посещаемости товара.

Также с помощью технологии обработки больших данных можно реализовать подбор статей по критериям пользователя.

1. Постановка задачи

Каждый день в поисковых системах составляют множество запросов определяемыми теми или иными критериями. Хорошо, когда пользователь знает какие-либо критерии, по которым он хочет получить информацию, а если нет, тогда ему предлагаются статьи, которые получаются в результате рекомендательных алгоритмов.

Основной задачей подбора статей является получения списка статей, которые будут наиболее интересны конкретному пользователю, на основе уже имеющихся данных об объектах или пользователе. Например, рассмотрим подборы на примере сервиса по предоставлению рецептов. Критерии выбора будут следующие:

- подбор по характеристикам пользователя (дата рождения, пол);
- подбор по геопозиции;
- подбор по уже просмотренным рецептам;
- подбор по часто используемым ингредиентам.

2. Решение задачи

Для реализации быстрых вычислений используется технология кластерных вычислений Apache Spark. Spark обрабатывает несколько запросов быстро и с минимальными издержками.

Spark Streaming реализует микропакетный подход (micro-batch), когда поток данных разбивается на небольшие пакеты временных интервалов. Абстракция Spark для потока представляет собой микро-пакет, содержащий несколько распределенных датасетов – RDD. Именно RDD является основным вычислительным примитивом Spark , над которым можно делать параллельные вычисления и преобразования[1].

Для решения поставленной задачи будем рассматривать подбор статей на примере сервиса по предоставлению рецептов.

На клиентской стороне получаем некоторые данные, например, идентификатор пользователя и отправляем его на сервер, где собираются данные о пользователе, о его сохраненных рецептах, если такие имеются. Результат обрабатывается с помощью набора модулей, каждый из которых использует спарк стриминг для обработки больших данных. Каждый модуль отвечает за поиск рецептов по определенным критериям (рис. 1):

• модуль заказов текущего пользователя – берем все просмотренные рецепты пользователя, выделяем из них наиболее встречаемые категории продуктов и по этим данным из всей библиотеки рецептов подбираем рецепты с такими же категориями;

• модуль пользовательских данных – берем данные тех пользователей, которые схожи с текущим по таким параметрам как пол, возраст, город. Затем у каждого из них находим сохраненные рецепты и ищем объединения, то есть рецепт которые есть у максимального количества пользователей;

• модуль определения стоимости – получаем список подобранных рецептов. И рассчитываем стоимость каждого блюда, сортируем по возрастанию цены;

• модуль популярности – из всей библиотеки рецептов получаем список тех рецептов, которые были чаще всего сохранены всеми пользователями.

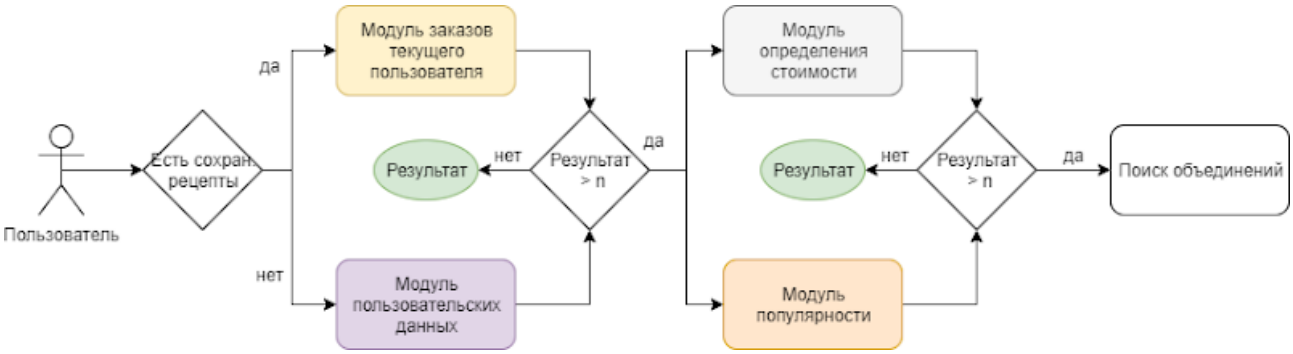


Рис. 1. Схема модулей

Так, например, у нас есть S рецептов, количество которых постоянно увеличивается и множество пользователей C, где каждому пользователю c необходимо подобрать необходимое количество рецептов.

Подбор рецептов в одном модуле можно представить следующим образом[2]:

$$\forall c \in C, \quad s'_c = \arg\max_{s \in S} u(c, s),$$

где C – множество пользователей, S – множество рецептов, u – функция, определяющая насколько рецепт s удовлетворяет некоторого пользователя c. Таким образом, необходимо выбрать такие рецепты $s' \in S$, при котором значения удовлетворенности для каждого пользователя $c \in C$ максимально.

Данная формула описывает подбор рецептов в каждом вышеперечисленном модуле. В результате получаем значения, максимально удовлетворяющие условиям в текущем модуле, для каждого пользователя. Объединение значений позволит определить искомый список значений, который возможно будет интересен для определенного пользователя.

В результате описанного выше анализа была спроектирована архитектура приложения, представленная на рис. 2.

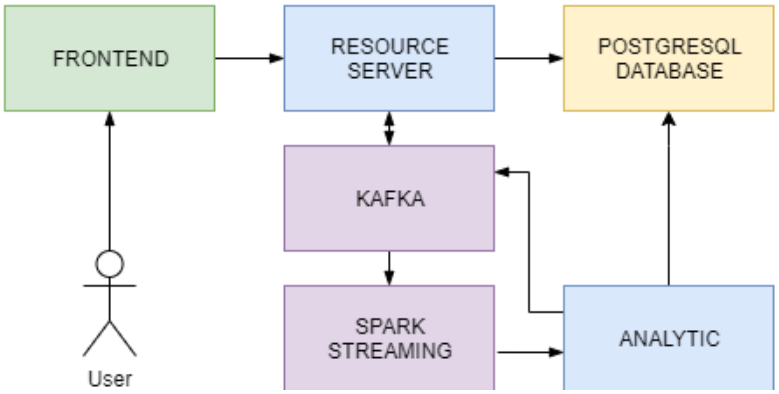


Рис. 2. Архитектура приложения

Заключение

В данной работе был приведен алгоритм подбора статей. В рамках работы были разработаны модули на Apache Spark для быстрой обработки больших данных, которые будут использоваться в дальнейшем для поиска рецептов.

Литература

1. Изучаем Spark: молниеносный анализ данных / Карау Х., Конвински Э., Венделл П., Захария М. – М.: ДМК Пресс, 2015. – 304 с.

2. Adomavicius G. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions / G. Adomavicius, A. Tuzhilin // IEEE Transactions on Knowledge and Data Engineering. – 2005. – Vol. 17, № 6. – P. 734–749.

Попова Алина Витальевна – магистрант 2-го года обучения кафедры МО ЭВМ Воронежского государственного университета. E-mail: popowa.alinochka@gmail.com

Трофименко Елена Владимировна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры МО ЭВМ Воронежского государственного университета.
E-mail: evtrof@gmail.com

УДК 519.72

СУММА ВОЗРАСТАЮЩИХ ГЕНЕРАТОРОВ В ФОРМЕ ЛОГАРИФМА ОТ ДРОБНО-ЛИНЕЙНЫХ ФУНКЦИЙ

Я. В. Попова

Воронежский государственный университет

Введение

Целенаправленный подход для формирования различных нечетких операций стал возможным благодаря введению треугольных норм и конорм [1–3]. Примерами таких операций являются следующие хорошо известные пары двойственных треугольных норм и конорм [2]:

$$T_0(x,y)=\frac{xy}{x+y-xy},\ S_{-1}(x,y)=\frac{x+y-2xy}{1-xy};$$

$$T_\alpha(x,y)=\frac{xy}{\alpha+(1-\alpha)(x+y-xy)}\ (\alpha>0),\ S_\beta(x,y)=\frac{(\beta-1)xy+x+y}{1+\beta xy}\ (\beta>-1);$$

$$T_P(x,y)=xy,\ \ S_P(x,y)=x+y-xy;$$

Важнейшим результатом данного направления является представление треугольных норм и конорм с помощью аддитивных генераторов. Данное представление связано со свойством ассоциативности бинарных операций [3, 4]. Известные параметрические семейства треугольных норм и конорм и соответствующие им аддитивные генераторы можно найти в [2]. Исследование треугольных норм и конорм в терминах их аддитивных генераторов лежит в основе ряда приложений, среди которых выделим следующие: конструирование различных нечетких алгебр; определение различных типов композиций нечетких отношений и видов транзитивности; определение специальных операций над нечеткими числами; определение операций агрегирования.

Заметим, что перечисленные выше операции относятся к классу рациональных функций. В [4] установлено, что их аддитивные генераторы представляются в виде дробно-линейной функции $\varphi(x)=\frac{ax+b}{cx+d}$, а также $\ln\varphi$ и $\operatorname{arctg}\varphi$. В [5] исследовались аддитивные генераторы φ, для них получены соответствующие треугольные нормы и конормы. В [6] рассмотрены возрастающие генераторы $\ln\varphi$.

Цель статьи заключается в исследовании параметров суммы генераторов из данного класса.

1. Основные определения и факты

Треугольной нормой (*t-нормой*) называется коммутативная, ассоциативная и монотонная бинарная операция $T:[0,1]\times[0,1]\to[0,1]$, такая, что для всех $x\in[0,1]$ выполняется ограничение $T(x,1)=x$.

Треугольная конорма (*t-конорма*) $S:[0,1]\times[0,1]\to[0,1]$ удовлетворяет тем же свойствам, но ограничение для всех $x\in[0,1]$ имеет вид $S(x,0)=x$.

Треугольная норма моделирует операции типа умножения (пересечение нечетких множеств, конъюнкция), а конорма – операции типа сложения (объединение нечетких множеств, дизъюнкция).

Известно, что непрерывная треугольная конорма представляется с помощью возрастающего генератора s в виде

$$S(x,y)=s^{(-1)}\big(s(x)+s(y)\big),$$

где $s^{(-1)}$ – псевдообратная функция, совпадающая на $[0,1]$ со своей обратной s^{-1}.

Возрастающим генератором называется строго возрастающая непрерывная функция $s:[0,1]\to[0,\infty)$, такая что $s(0)=0$.

В [6] были определены ограничения на коэффициенты возрастающих генераторов в форме логарифма от дробно-линейной функции. Имеет место следующее утверждение: Функция $\varphi_{\alpha,\beta}(x)=\ln\left(\dfrac{\alpha x+1}{\beta x+1}\right)$, где $\alpha\neq 0$ и $\beta\neq 0$, является возрастающим генератором при следующих ограничениях на параметры: $-1<\beta<\alpha<0$; $-1<\beta<0<\alpha$; $0<\beta<\alpha$.

2. Сумма возрастающих генераторов

Пусть заданы два возрастающих генератора $\varphi_1(x)=\ln\dfrac{\alpha_1 x+1}{\beta_1 x+1}$ и $\varphi_2(x)=\ln\dfrac{\alpha_2 x+1}{\beta_2 x+1}$ со следующими ограничениями на параметры: $-1<\beta_1<\alpha_1<0$, $-1<\beta_1<0<\alpha_1$, $0<\beta_1<\alpha_1$ и $-1<\beta_2<\alpha_2<0$, $-1<\beta_2<0<\alpha_2$, $0<\beta_2<\alpha_2$ соответственно. Рассмотрим функцию, которая является их суммой, и определим ограничения, при выполнении которых сумма возрастающих генераторов является возрастающим генератором.

Пусть $\gamma(x)=\ln\dfrac{\alpha_1 x+1}{\beta_1 x+1}+\ln\dfrac{\alpha_2 x+1}{\beta_2 x+1}$, тогда по свойству логарифмической функции получим

$$\gamma=\ln\left(\frac{\alpha_1 x+1}{\beta_1 x+1}\cdot\frac{\alpha_2 x+1}{\beta_2 x+1}\right) \tag{1}$$

После преобразований формула (1) будет иметь вид:

$$\gamma=\ln\frac{\alpha_1\alpha_2 x^2+(\alpha_1+\alpha_2)x+1}{\beta_1\beta_2 x^2+(\beta_1+\beta_2)x+1} \tag{2}$$

Для того, чтобы функция γ была функцией вида $\varphi_{\alpha,\beta}(x)=\ln\dfrac{\alpha x+1}{\beta x+1}$, необходимо потребовать выполнение следующих условий: $\begin{cases}\alpha_1\alpha_2=0\\ \beta_1\beta_2=0\end{cases}$, т. е. один из коэффициентов α_1,α_2 и β_1,β_2 должен быть равен 0. Тогда получим следующие ситуации:

$$1)\ \begin{cases}\alpha_1=0,\\ \beta_1=0;\end{cases}\quad 2)\ \begin{cases}\alpha_1=0,\\ \beta_2=0;\end{cases}\quad 3)\ \begin{cases}\alpha_2=0,\\ \beta_1=0;\end{cases}\quad 4)\ \begin{cases}\alpha_2=0,\\ \beta_2=0,\end{cases}$$

которые порождают частные случаи исходных генераторов φ_1 и φ_2.

В случае 1) $\gamma=\ln\dfrac{\alpha_2 x+1}{\beta_2 x+1}$, а это в свою очередь $\varphi_2(x)$, следовательно γ является возрастающим генератором с теми же ограничениями на параметры, что и для $\varphi_2(x)$.

В случае 2) необходимо исследовать ограничения на параметры для функции

$$\gamma=\ln\frac{\alpha_2 x+1}{\beta_1 x+1}. \tag{3}$$

Учитывая ограничения на параметры для возрастающих генераторов $\varphi_1(x)$ и $\varphi_2(x)$, получим следующие ограничения на параметры:

a) $\begin{cases}-1<\alpha_2<0,\\ -1<\beta_1<0;\end{cases}$ b) $\begin{cases}-1<\alpha_2<0,\\ \beta_1>0;\end{cases}$ c) $\begin{cases}\alpha_2>0,\\ -1<\beta_1<0;\end{cases}$ d) $\begin{cases}\alpha_2>0,\\ \beta_1>0.\end{cases}$

Найдем область определения данной функции $\begin{cases}\alpha_2 x+1\neq 0,\\ \beta_1 x+1\neq 0,\\ (\alpha_2 x+1)(\beta_1 x+1)>0.\end{cases}$ Функция (3) имеет две вертикальных асимптоты $x_1=-\frac{1}{\alpha_2}$, $x_2=-\frac{1}{\beta_1}$ и горизонтальную асимптоту $y=\ln\frac{\alpha_2}{\beta_1}$.

Производная функции (3) определяется формулой $\gamma'=\frac{\alpha_2-\beta_1}{(\alpha_2 x+1)(\beta_1 x+1)}$. Тогда (3) с учетом области определения будет возрастающим генератором при условии, что $\alpha_2>\beta_1$. Таким образом случай b) не подходит под данное условие.

Утверждение 1. *Сумма $\varphi_1+\varphi_2$ возрастающих генераторов $\varphi_1(x)=\ln\frac{1}{\beta_1 x+1}$ и $\varphi_2(x)=\ln\alpha_2 x+1$, где $\alpha_2\neq 0$ и $\beta_1\neq 0$ является возрастающим генератором при следующих ограничениях на параметры: $-1<\beta_1<\alpha_2<0$, $-1<\beta_1<0<\alpha_2$, $0<\beta_1<\alpha_2$.*

В 3) случае получим

$$\gamma=\ln\frac{\alpha_1 x+1}{\beta_2 x+1}. \tag{4}$$

Нам необходимо исследовать различные ограничения на параметры. С учетом ограничений на возрастающие генераторы $\varphi_1(x)$ и $\varphi_2(x)$ получим следующие ограничения на параметры:

a) $\begin{cases}-1<\alpha_1<0,\\ -1<\beta_2<0;\end{cases}$ b) $\begin{cases}-1<\alpha_1<0,\\ \beta_2>0;\end{cases}$ c) $\begin{cases}\alpha_1>0,\\ -1<\beta_2<0;\end{cases}$ d) $\begin{cases}\alpha_1>0,\\ \beta_2>0.\end{cases}$

Найдем область определения данной функции $\begin{cases}\alpha_1 x+1\neq 0,\\ \beta_2 x+1\neq 0,\\ (\alpha_1 x+1)(\beta_2 x+1)>0.\end{cases}$ Функция (3) имеет две вертикальных асимптоты $x_1=-\frac{1}{\alpha_1}$, $x_2=-\frac{1}{\beta_2}$ и горизонтальную асимптоту $y=\ln\frac{\alpha_1}{\beta_2}$.

Производная функции (3) определяется формулой $\gamma'=\frac{\alpha_1-\beta_2}{(\alpha_1 x+1)(\beta_2 x+1)}$. Тогда (3) с учетом области определения будет возрастающим генератором при условии, что $\alpha_1>\beta_2$. Таким образом случай b) не подходит под данное условие.

Таким образом, доказано следующее

Утверждение 2. *Сумма двух возрастающих генераторов $\varphi_1(x)=\ln\alpha_1 x+1$ и $\varphi_2(x)=\ln\frac{1}{\beta_2 x+1}$, где $\alpha_1\neq 0$ и $\beta_2\neq 0$ является возрастающим генератором при следующих ограничениях на параметры: $-1<\beta_2<\alpha_1<0$, $-1<\beta_2<0<\alpha_1$, $0<\beta_2<\alpha_1$.*

Рассмотрим 4) случай, тогда $\gamma=\ln\frac{\alpha_1 x+1}{\beta_1 x+1}$, а это в свою очередь $\varphi_1(x)$, следовательно γ является возрастающим генератором с теми же ограничениями на параметры, что и для $\varphi_1(x)$.

На рис. 1 приведены графики возрастающих генераторов $\varphi_1(x)=\ln\frac{1}{-0,7x+1}$ и $\varphi_2(x)=\ln(2x+1)$ и их суммы $\gamma=\ln\frac{2x+1}{-0.7x+1}$.

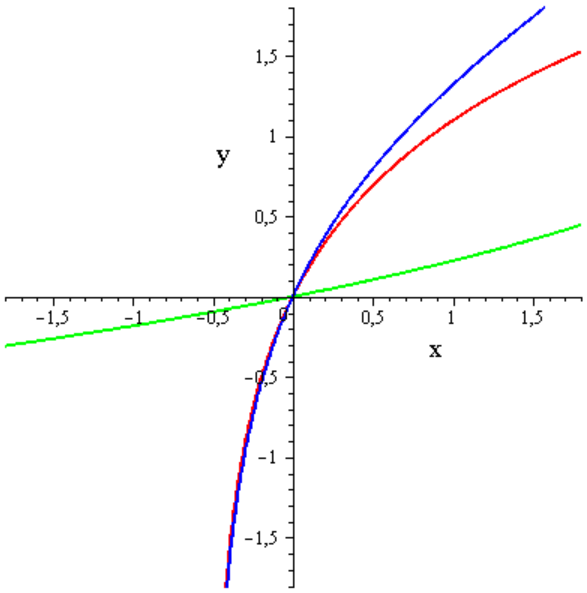


Рис. 1. Графики возрастающих генераторов $\varphi_1(x)$ *(зеленый) и* $\varphi_2(x)$ *(красный), а также их суммы* γ *(синий)*

Заключение

В данной статье введена и исследована сумма возрастающих генераторов в форме логарифма от дробно-линейных функций при условии, что мультипликативная константа равна единице. А также определены ограничения на коэффициенты функции γ.

Литература

1. *Klement E. P., Mesiar R., Pap E.* Triangular norms. Position paper I: basic analytical and algebraic properties // Fuzzy Set and Systems. – 2004. – 143. P. 5–26.

2. *Klement E. P., Mesiar R., Pap E.* Triangular norms. Position paper II: general constructions and parameterized families, Fuzzy Set and Systems. 2004. – 145. P. 439–454.

3. *Klement E. P., Mesiar R., Pap E.* Triangular norms. Position paper III: continuous t-norms // Fuzzy Set and Systems. – 2004. – 145. P. 439–454.

4. *Леденева Т. М.* Некоторые аспекты представления нечетких операторов отношением двух многочленов / Т. М. Леденева // Известия ВУЗов. Математика. – 1997. – С. 33–40.

5. *Ledeneva T. M.* Additive generators of fuzzy operations in the form of linear fractional functions, Fuzzy Set and Systems. – https://doi.org/10.1016/j.fss.2019.03.005.

6. *Леденева Т. М.* Возрастающие генераторы в форме логарифма от дробно-линейных функций / Т. М. Леденева, Я. В. Попова // Актуальные проблемы прикладной математики, информатики и механики : сб. тр. Междунар. науч.-техн. конф. (Воронеж, 11–13 ноября 2019 г.) : электронный ресурс. – Воронеж, 2019. – С. 2–5.

Попова Яна Викторовна – студентка 4-го курса кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: popova.jana2013@yandex.ru

Леденева Татьяна Михайловна (научный руководитель) – д-р техн. наук, профессор, зав. кафедрой вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail ledeneva-tm@yandex.ru

УДК 004.932.2

МОБИЛЬНЫЙ МОДУЛЬ SMARTWALL

М. А. Принев

Воронежский государственный университет

Введение

В настоящее время все более востребованными становятся программные продукты с использованием компьютерного зрения (определение лиц на фотографиях, управление компьютером при помощи жестов, распознавание объектов на изображении, датчики движения, системы технического зрения и т. д.), а также все большую популярность приобретает использование интерактивных поверхностей (сенсорные экраны, панели, интерактивные сенсорные поверхности и т. д.). Компьютерное зрение (Computer Vision, CV), в том числе машинное зрение (Machine Vision, MV) – это автоматическая фиксация и обработка изображений, как неподвижных, так и движущихся объектов при помощи компьютерных средств. В общем случае, системы CV состоят из фото- или видеокамеры, а также компьютера, на котором работают программы обработки и анализа изображений. CV – быстро растущая область цифровых технологий, которая затрагивает многие стороны повседневной жизни [1].

В течение последних двух лет ведется НИР, основной целью которой является создание на основе разработанного автором метода сегментации зашумленных изображений с плавающим порогом бинаризации программного обеспечения SmartWall, позволяющего использовать компьютерное зрение для применения любой поверхности в качестве интерактивного элемента без использования сенсорных технологий.

Использование метода в программном обеспечении позволяет создавать интерактивные области на поверхностях любого цвета и текстуры, а также изображениях, полученных с помощью проектора без использования сенсорных технологий, что по предварительной оценке позволит снизить стоимость неэлектронных гаджетов примерно на 60–70 % по сравнению с их электронными аналогами. Речь идет об использовании программных продуктов, созданных на основе ПО SmartWall [1, 3], для разработки гораздо более дешевого (по предварительной оценке, до 60 %) оборудования, применяемого в сенсорных комнатах для детей. Использование любых поверхностей в качестве интерактивного несенсорного элемента позволит создать бюджетную альтернативу сенсорным доскам, а также позволит осуществлять выпуск одноразовых неэлектронных гаджетов в случаях, когда их серийный выпуск нецелесообразен.

Разработка мобильного модуля SmartWall позволит решить эргономические проблемы, возникающие из-за ограничений, связанных с малой длиной провода, соединяющего веб-камеру с компьютером, а также обеспечить удаленный доступ пользователям.

1. Создание программных средств, обеспечивающих совместную работу веб-камеры и одноплатного компьютера Raspberry Pi

Программные средства, обеспечивающие совместную работу веб-камеры и одноплатного компьютера Raspberry Pi, были разработаны на языке программирования Java 8 [2, 4], в среде разработки IntelliJ IDEA, для ОС Linux.

Разработанные программные средства предназначены для видео-захвата изображения с последующей обработкой по методу сегментации зашумленных изображений с плавающим

порогом бинаризации, разработанному автором проекта. Программные средства выполняют следующие функции:

• запуск ПО SmartWall на компьютерах, оснащенных ОС Linux;

• реализацию возможности удаленной настройки ПО SmartWall, позволяющей увеличить мобильность системы;

• обеспечение стабильности и повышение быстродействия ПО SmartWall, не более 3 с при различных уровнях освещения в диапазоне 30–1000 лк;

• решение эргономических проблем, возникающих из-за ограничений, связанных с малой длиной провода, соединяющего веб-камеру с компьютером.

В ходе разработки экспериментальным путем было установлено, что повышение ресурсоемкости процессов на одноплатном компьютере Raspberry Pi, приводит к его нагреванию и потере до 30 % мощностных характеристик, таких как частота процессора и скорость обработки данных, что ведет к соответствующему уменьшению быстродействия ПО SmartWall. Результатом эксперимента стало принятие решения о целесообразности схемы работы программного обеспечения, представленной на рис. 1.

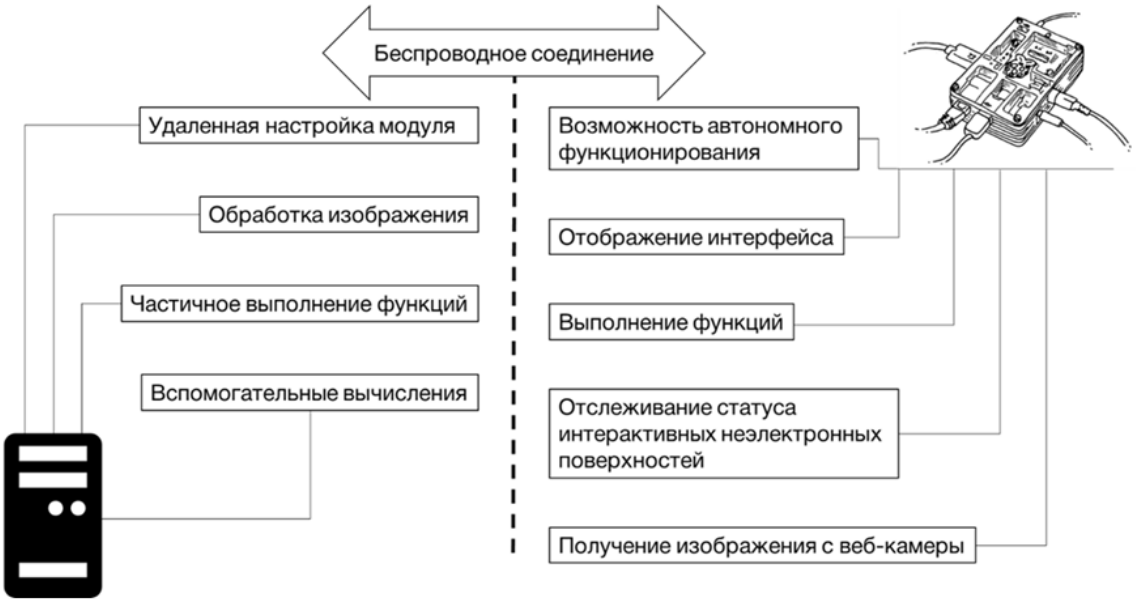


Рис. 1. Схема принципа работы модуля

2. Решение конструктивных проблем соединения веб-камеры с одноплатным компьютером Raspberry Pi для создания модуля удаленного доступа

В ходе второго этапа НИР были решены конструктивные проблемы соединения веб-камеры с одноплатным компьютером Raspberry Pi для создания модуля удаленного доступа, а именно разработка оптимальной конструкции соединения веб-камеры с одноплатным компьютером в компактном корпусе с целью обеспечения стабильной работы ПО SmartWall. Первоначальная конструкция мобильного модуля SmartWall включала в себя элементы, представленные в табл. 1.

Внешний вид мобильного модуля SmartWall в сборе представлен на рис. 2.

Таблица 1

Элементы конструкции модуля SmartWall

№ п/п	Элемент конструкции	Основные характеристики	Статус элемента
1	Raspberry Pi 3 model B	Количество ядер процессора: 4, Частота процессора: 1200 МГц, Размер оперативной памяти: 1024 МБ	Приобретен
2	Блок питания, Micro USB	Выходное напряжение: 5V, Максимальный выходной ток: 2A	Приобретен
3	Camera Module v1	Габаритные размеры: 25 × 24 × 9 мм, Разрешение: 5 Megapixels	Приобретен
4	Карта памяти microSDHC	Объем памяти: 8 ГБ, Класс 10	Приобретен
5	Набор радиаторов	Материал: медь, Габаритные размеры: 15 × 15 × 1 мм, Количество: 3 шт.	Приобретен
6	Корпус	Материал: акрил, Габаритные размеры: 95 × 65 × 35 мм, Наличие посадочных и крепежных отверстий	Разработан
7	Регулируемое крепление камеры	Материал: прессованный картон, h = 3 мм, Габаритные размеры: 40 × 35 × 30 мм	Разработан

Рис. 2. Мобильный модуль SmartWall

3. Тестирование работы модуля удаленного доступа в реальных условиях

Было проведено тестирование модуля удаленного доступа в реальных условиях с целью определить уровень влияния нагревания конструкции при работе модуля на основные показатели производительности. Для тестирования использовались программные средства, раз-

работанные автором проекта на языке программирования Python. В процессе тестирования замерялась рабочая температура частота работы процессора при различной нагрузке и продолжительности работы модуля. Результаты тестирования модуля SmartWall представлены в табл. 2.

Таблица 2

Результаты тестирования работы модуля с пассивным охлаждением

Время работы модуля, мин	Частота работы процессора, MHz	Температура ЦПУ, °C
0	1200	38
3	1200	41
6	1100	46
9	900	55
12	850	61
15	730	63

Результаты тестирования показали, что пассивного охлаждения модуля недостаточно для его стабильной и продуктивной работы. По результатам тестирования было принято решение добавить в конструкцию модуля элемент активного охлаждения, а именно процессорный кулер для Raspberry PI 3. Была проведена коррекция конструкции модуля, заключающаяся в расчете и реализации посадочных мест на корпусе модуля в соответствии с конструкцией и крепежными габаритами кулера. После завершения работ по оптимизации конструкции модуля было проведено повторное тестирование, результаты которого представлены в табл. 3.

Таблица 3

Результаты тестирования работы модуля с активным охлаждением

Время работы модуля, мин	Частота работы процессора, MHz	Температура ЦПУ, °C
0	1200	38
3	1200	38
6	1200	40
9	1200	43
12	1100	45
15	1100	46

Результаты повторного тестирования показали, что добавление в конструкцию модуля системы активного охлаждения позволило улучшить тестируемые показатели в среднем на 30 %. Конструкция мобильного модуля SmartWall, оптимизированная по результатам тестирования, представлена на рис. 3.

Заключение

В рамках второго этапа НИР в соответствии с календарным планом были проведены следующие работы:

1) Создание программных средств, обеспечивающих совместную работу веб-камеры и одноплатного компьютера Raspberry Pi. Разработанные на языке программирования Java 8, в среде разработки IntelliJ IDEA программные средства выполняют функции, необходимые для корректной работы ПО SmartWall (адаптация программного обеспечения для ОС Linux,

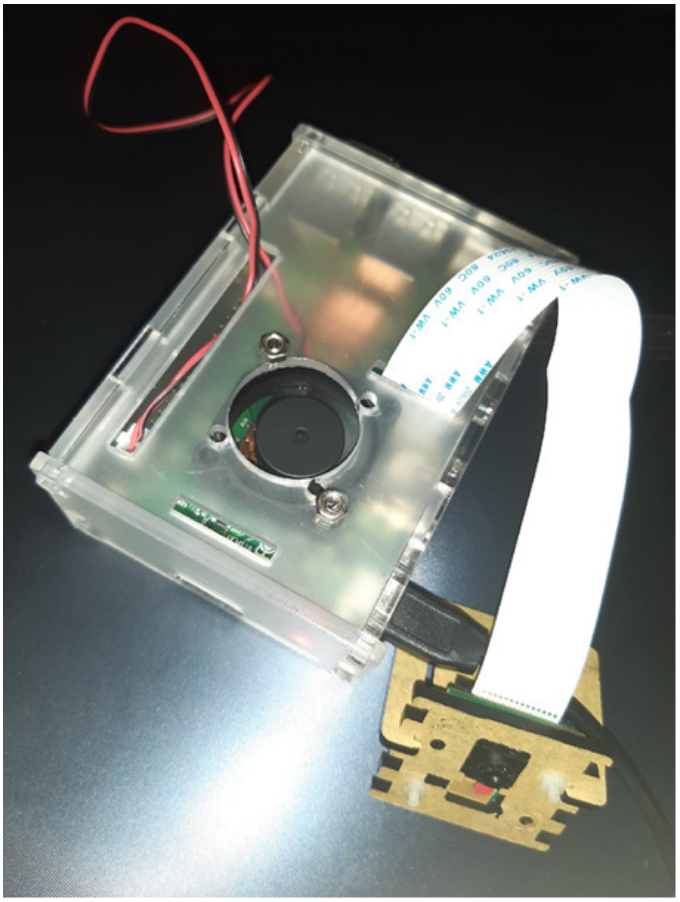


Рис. 3. Мобильный модуль SmartWall с системой активного охлаждения

обеспечение необходимого быстродействия, а также реализация возможности удаленной настройки ПО SmartWall). По результатам проведенных экспериментов была оптимизирована схема принципа работы программного обеспечения.

2) Решение конструктивных проблем соединения веб-камеры с одноплатным компьютером Raspberry Pi для создания модуля удаленного доступа, а именно разработка оптимальной конструкции соединения веб-камеры с одноплатным компьютером в компактном корпусе с целью обеспечения стабильной работы ПО SmartWall, включающая в себя как приобретенные, так и разработанные элементы конструкции, такие как одноплатный компьютер Raspberry Pi 3B, камеру с регулируемым креплением, карту памяти, блок питания, корпус и средства пассивного и активного охлаждения.

3) Тестирование работы модуля удаленного доступа в реальных условиях, доработка модуля удаленного доступа по результатам его тестирования. Результаты проведенного тестирования показали, что модуль нуждается в средствах активного охлаждения, добавление в конструкцию процессорного кулера позволило до 30% снизить потери производительности и быстродействия работы программного обеспечения, что было также было подтверждено серией соответствующих тестов.

Благодарности

НИР по проекту проводится в рамках гранта от ФГБУ «Фонд содействия развитию малых форм предприятий в научно-технической сфере» (Фонд содействия инновациям), полученного автором в качестве победителя программы УМНИК 2018 г.

Литература

1. *Тропченко, А. Ю.* Методы вторичной обработки изображений и распознавания объектов. Учебное пособие / А. Ю. Тропченко – СПб: СПбГУ ИТМО, 2012. – 52 с.

2. *Окулов, С. М.* Основы программирования / С. М. Окулов. – 5-е изд., испр. – М : БИНОМ. Лаборатория знаний, 2010. – 440 с.

3. Visual C# [Электронный ресурс]. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/kx37x362%28v=vs.120%29

4. Документация по Java Platform Standard Edition 8 [Электронный ресурс]. – Режим доступа: https://docs.oracle.com/javase/8/docs/

Принев Мечислав Александрович – студент 3-го курса кафедры ПОиАИС Воронежского государственного университета. E-mail: metshislav@rambler.ru

Воронина Ирина Евгеньевна (научный руководитель) – д-р. техн. наук, доцент, проф., профессор кафедры ПОиАИС Воронежского государственного университета. E-mail: irina.voronina@gmail.com

УДК 004.925

ИССЛЕДОВАНИЕ СИСТЕМ СГЛАЖИВАНИЯ И РЕАЛИЗАЦИЯ ВРЕМЕННОГО СГЛАЖИВАНИЯ

Н. А. Псарев

Воронежский государственный университет

Введение

При растеризации 3D-графики, из-за ограниченного разрешения компьютерных экранов, некоторые линии и кривые будут деформироваться. Это явление называется алиасинг (англ. aliasing). Постоянно растущий спрос на более реалистичную графику привел к созданию новых методов и алгоритмов для скрытия этих артефактов, не обязательно используя экраны с более высоким разрешением. Сглаживание стало одним из важнейших методов повышения качества изображения в графическом программном обеспечении. Эта область была исследована в течение многих лет, и постоянно разрабатывались новые подходы. Эта статья направлена на анализ различных разработанных методов сглаживания и реализацию метода временного сглаживания, совместимого с движками отложенного затенения и среди прочих преимуществ обладающего таким же качеством, как и мультисэмплинг (англ. multisampling).

1. Постановка задачи

Необходимо проанализировать состояние разработок в области графических вычислений в отношении сглаживания и исследовать проблемы, связанные с алиасингом в 2D и 3D-графике, эволюцию решений и текущие разработками. Также целью является интегрировать метод временного сглаживания для улучшения качества изображения графического движка. Конечный результат должен иметь качество, эквивалентное качеству других более традиционных решений, таких как суперсэмплинг (англ. supersampling)., но с гораздо меньшим влиянием на производительность, например, с традиционными постобработками алиасинга.

2. Материалы и методы

2.1. Типы алиасинга

Алиасинг – это деформация определенных графических элементов при растеризации и воплощении на экране с конечным разрешением [1]. Обычно геометрические ребра, которые имеют углы, не совпадающие идеально с пикселями, представляют собой резкие пильные ребра, которые не так точно соответствуют сцене, как это было бы в реальности.

Алиасинг по границам геометрии – самый распространенный и известный вид алиасинга с точки зрения графических вычислений. Этот эффект возникает на краях геометрии сцены. Эффект проявляется в виде зубчатых краев. Проявление особенно заметно в диагональных линиях и кривых, так как в плоских линиях (как горизонтальных, так и вертикальных) они совпадают с массивом пикселей экрана. Это связано с сэмплингом, который используется во время растеризации сцены (рис. 1).

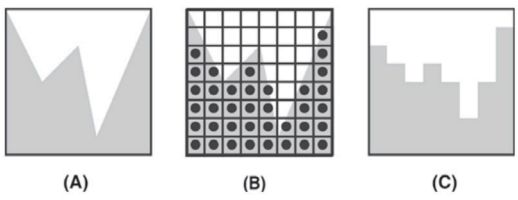


Рис. 1. Сэмплинг

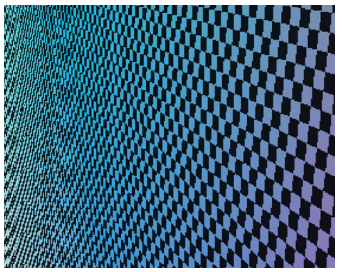


Рис. 2. Узор муара

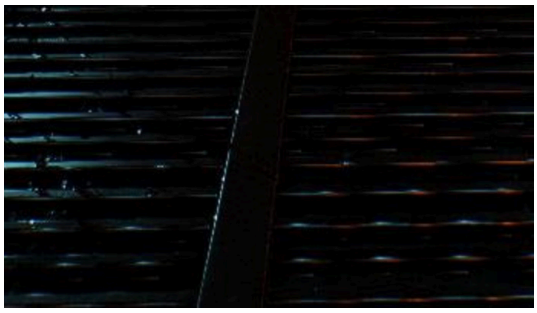


Рис. 3. Алиасинг отражения в видеоигре "DOOM (2016)"

Внутри текстур также может быть алиасинг. Подобно тому, как это происходит по краям геометрии, при быстром переключении с одного цвета на другой из-за ограниченного выборки могут возникнуть эти же дефекты.

Этот тип алиасинга не ограничивается рендерингом 3D-графики в реальном времени, но его можно увидеть даже на изображениях с высоким разрешением. В этих случаях, в зависимости от текстуры, можно увидеть узор муара, как на рис. 2.

Отраженное освещение – классическая модель освещения, используемая в OpenGL, состоящая из компонентов окружающего, рассеянного и отраженного света [8]. Отраженное освещение возникает, когда свет попадает непосредственно на материал. Свойства объекта будут диктовать, насколько ярко он выглядит.

Расчет этих отражений может содержать алиасинг, так же, как и текстуры. На рис. 3 это можно рассматривать в отражении металлического пола страдающий от той же проблемы, что и при алиасинге внутри текстур.

2.2. Типы сглаживания

Сглаживание – это набор техник, цель которых состоит в том, чтобы скрыть или устранить недостатки, отмеченные выше.

Фильтрация

В растровой графике (двумерной) сглаживание выполняется с помощью фильтров. Чтобы понять фильтрацию в изображениях, необходимо думать о пикселях не как о маленьких квадратах на экране, а скорее, как о выборках функции (обычно 3 сэмпла, по одному для каждого канала RGB) [2]. Фильтры (или ядра) преобразуют функции, чтобы попытаться сгладить внезапные изменения цвета.

На рис. 4 показано сравнение различных способов, которыми каждый фильтр интерполирует значения изображения с помощью одного канала. Каждая точка представляет собой образец функции, которая будет отображаться на экране.

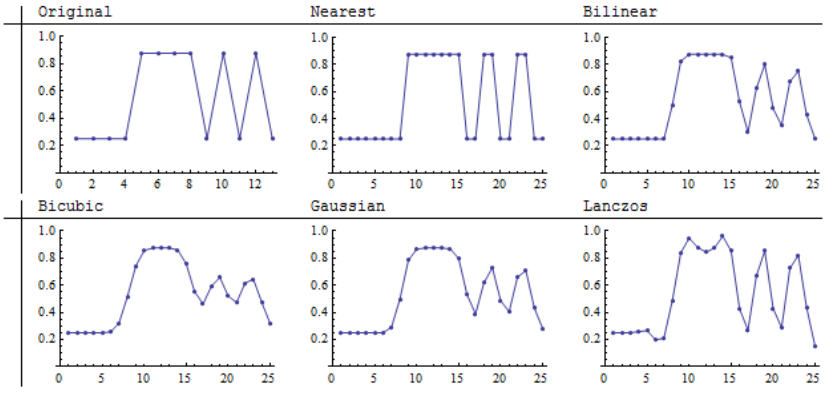


Рис. 4. Сравнение интерполяции с использованием разных фильтров [7]

Сглаживание суперсэмплированием

Суперсэмплирование состоит в минимизации сглаживания с использованием изображений с разрешением выше, чем то, которое используется для их визуализации (пространственное сглаживание). Каждый пиксель на экране будет представлен более чем одним пикселем в изображении. Для конечного цвета значение цветов нескольких пикселей обычно усредняется [3].

По сравнению с другими методами сглаживания суперсэмплинг позволяет корректировать алиасинг текстур (рис. 2) и алиасинг отражений (см. рис. 3), а также геометрическое сглаживание. Это происходит потому, что подсэмплы взяты со всей сцены.

Сглаживание мультсэмплированием

Мультисэмплинг – это оптимизация суперсэмплинга. Субпиксели все еще используются, но вместо вычисления цвета для каждого субпикселя, цвет вычисляется только один раз на пиксель. После расчета цвета, вычисляются буфер трафарета и буфер глубины с субпикселями, и в зависимости от количества субпикселей, содержащихся в треугольнике, будет использоваться процент от вычисленного значения цвета. Этот метод значительно снижает влияние на производительность приложения, так как текстура должна быть доступна только один раз на пиксель, но решает только алиасинг по краям геометрии [4][5].

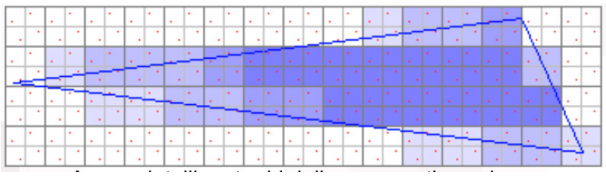


Рис. 5. Мультисэмплирование

Поскольку расчет конечного цвета каждого пикселя производится в пиксельном шейдере после расчета g-буфера (рис. 5), то при выполнении мультисэмплирования разрешения не вся необходимая информация готова. В OpenGL 3.2 и Direct3D 10.1 это изменилось благодаря явному мультисэмплингу [10].

Временное сглаживание

Временное сглаживание является одним из видов постобработки сглаживания, но оно амортизирует вычислительные затраты при использовании нескольких выборок в нескольких кадрах. Оно создает некоторые сложности при выборе образцов в предыдущих кадрах на нестатических изображениях. Его реализация и результаты более подробно описываются в следующих главах.

3. Реализация временного сглаживания

Система временного сглаживания была интегрирована в 3D-графический движок, который использует OpenGL в качестве графического API. Его реализация была осуществлена с использованием шейдеров GLSL и кода C++ 17. Никакие функции сглаживания, которые автоматически предоставляет OpenGL, не использовались, например, использование текстур GL_TEXTURE_2D_MULTISAMPLE. Для этого движку необходимо уметь сохранять результат предыдущего кадра, слегка перемещать камеру по каждому кадру (создавая уникальные подсэмплы в каждом кадре), вычислять смещение камеры, смешивать цвета из предыдущего кадра в текущий кадр и, наконец, применять фильтр резкости.

3.1. Буфер истории

В связи с необходимостью объединения нескольких текстур из предыдущих кадров, для расчета сглаживания кадра n необходимо иметь доступ к кадру $n-1$. Даже при использовании более чем 2 подсэмплов необходимо сохранить только последнюю, если она уже включает

сглаживание. Также можно сохранить n кадров для n подсэмплов и объединить их в пиксельный шейдер, но это занимает гораздо больше памяти и нецелесообразно. Для этого необходимо создать FBO и сохранить результат текстуры в виде переменной, которая во время следующего кадра будет передаваться шейдеру в виде однородного sampler2D, и таким образом иметь возможность доступа к его пикселям.

3.2. Дрожание

Прежде всего необходимо выбрать подсэмплы для использования в каждом кадре. Важно, что вместо вычисления нескольких подсэмплов в каждом кадре временное сглаживание вычисляет один сэмпл на кадр. Каждый кадр будет изменять подсэмпл для расчета. Поскольку на каждый кадр вычисляется только один подсэмпл, влияние на производительность будет намного меньше, чем при традиционном суперсэмплировании.

Для выбора подсэмплов можно использовать сетки, рассмотренные ранее в разделе о суперсэмплировании [11]. Тем не менее, на практике чаще используются последовательности Холтона (рис. 6).

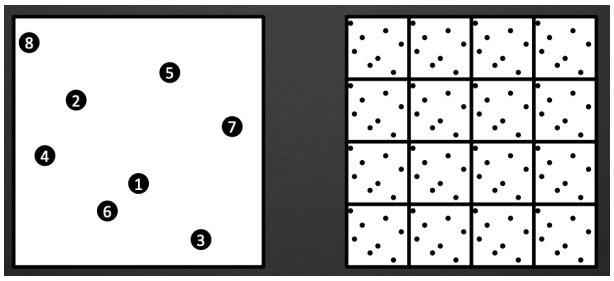


Рис. 6. Дрожание с последовательностью Холтона

Последовательности Холтона обеспечивают низкое расхождение, что означает, что никакие кластеры не будут созданы ни в пространстве, ни во времени. Это обеспечивает очень высокое качество подсэмпла, поэтому качество сглаживания может быть очень высоким. При хорошей реализации, результаты будут лучше, чем аппаратное сглаживание, реализованное до сих пор [11].

После того, как последовательность Холтона была вычислена в каждом кадре, матрица MVP будет изменена, чтобы переместить сэмпл в место подсэмпла, с которым он соприкасается в этом кадре. Это место будет меняться каждый раз, когда создается новый кадр.

На текущий момент, если посмотреть на изображение, можно все равно заметить сглаживание, и изображение будет казаться слегка дрожащим.

3.3. Вектора движения

Вторым шагом в реализации временного сглаживания является создание буфера скорости. Этот буфер будет содержать текстуру fp16 с двумя цветовыми каналами (красным и зеленым), где каждый пиксель представляет вектор.

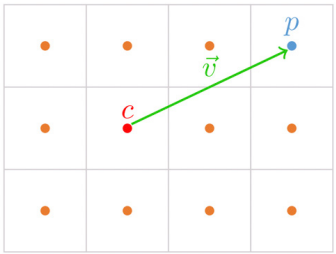


Рис. 7. $\overrightarrow{CP}$

Этот вектор представляет, насколько пиксель переместился из текущего кадра (n) в предыдущий кадр ($n-1$). Каждый пиксель определяется его позициями x и y. Для вычисления буфера скорости используется текущее положение $C = x_n y_n$ и предыдущее положение $P = x_{n-1} y_{n-1}$. $\overrightarrow{CP}$ хранится в буфере скорости, как показано на рис. 7.

Чтобы создать буфер скорости в динамической сцене, необходимо знать не только движение камеры в предыдущем кадре, но и то, как двигались объекты. Это усложняется, если существуют полупрозрачные объекты, такие как окна, или отражающие свет, такие как зеркала.

Для реализации выбран алгоритм, описанный в главе 27 GPU Gems 3 [6]. Необходимо только использовать пиксельный шейдер вместе с буфером истории, буфером глубины и матрицами VP_n и VP_{n-1}. Преимущество этого алгоритма заключается в том, что он не сильно зависит от реализации движка. Nvidia включает библиотеку, которая выполняет эти вычисления в PostWorks SDK [12] (используется для реализации TXAA).

Другая возможность заключается в вычислении скорости во время второго прохода вершинного шейдера каждого полигона, но в зависимости от его реализации она может быть несколько медленнее. Тем не менее это должно привести к более точным векторам и иногда это будет необходимо.

3.4. Сочетание текстур

Как только векторы движения были вычислены, комбинация текстур буфера истории и текущего кадра становится простой для выполнения. Достаточно будет совместить обе текстуры пиксель за пикселем в соответствующем месте, используя линейную интерполяцию. Значение 5 % часто используется в коммерческих [9] видеоиграх.

Если используется размытие, то рекомендуется использовать динамический коэффициент линейной интерполяции, зависящий от локального контраста пикселя с окружающими его пикселями.

3.5. Ограничение соседства

Рис. 8. Эффект призрака [11]

Векторы движения иногда недостаточно точны, чтобы их можно было использовать самостоятельно. Если объект перекрывает другой во время смены кадра, результирующий пиксель будет иметь неправильный цвет, так как текстуры из двух разных объектов будут объединены. Это может привести к эффекту, называемому призраком (рис. 8).

Чтобы исправить это, можно использовать ограничение соседства. Каждый пиксель сравнивается со значением соседних пикселей. Если изменение цвета больше или меньше, чем у соседних пикселей в буфере истории, то значение результирующего пикселя отбрасывается.

3.6. Ядро резкости

Из-за характера временного сглаживания окончательное изображение может оказаться слишком гладким. Края, возможно, окажутся несколько нечеткими. Этот эффект может не понравиться пользователю. Одним из возможных решений является использование ядра резкости, которое выделяет края геометрии.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

4. Результаты

Ниже приведены некоторые результаты внедрения временного сглаживания.

Рис. 9 содержит изображение, созданное движком без изменений. Значительное количество сглаживания можно увидеть на краю куба, который слегка наклонен. При активации временного сглаживания, как показано на рис. 10 и рис. 11, наблюдается заметное улучшение качества краев геометрии. При активации большего количества сэмплов, качество увеличивается.

Временное сглаживание также улучшает сглаживание отражений. Сравнивая рис. 9 с рис. 10, можно увидеть более мягкую кромку, также внутри геометрии. Эта проблема не может быть решена с помощью традиционного сглаживания мультисэмплирования.

Рис. 9. Сглаживание отключено

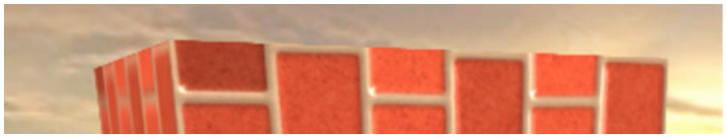


Рис. 10. Временное сглаживание с 8 сэмплами

Рис. 11. Временное сглаживание с 16 сэмплами

5. Заключение

Для достижения превосходных результатов в отношении качества изображения необходимо внедрить надежную систему сглаживания, способную решать различные аспекты сглаживания. Тот подход, который был применен в последние годы, точно отражает эту реальность. От перехода к использованию мультисэмплирования и ускорению его аппаратным обеспечением, к использованию систем постобработки, полностью реализованных в программном обеспечении, переходя через гибридные методы, достигнув временного сглаживания.

Временное сглаживание решает большую часть проблем сглаживания, таких как геометрические границы, текстуры, зеркальные отражения и часть конвейера (векторы движения), которые также могут быть повторно использованы для реализации размытия движения, которое приводит к сглаживанию движения. В дополнение к повторному использованию конвейера для решения задачи сглаживания движения, он может быть использован для других визуальных эффектов, таких как преграждение окружающего света в экранном пространстве, которые привносят еще больше реализма в сцены. Его совместимость и производительность, как и системы постобработки, делают его идеальным кандидатом для систем, работающих в режиме реального времени, таких как видеоигры.

В будущем предполагается расширить систему, включив в нее такие улучшения, как поддержка динамических сцен, прозрачных объектов, отражений, размытия движения, преграждение окружающего света в экранном пространстве и т. д.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Mitchell, D. P.* Reconstruction Filters in Computer-graphics / D. P. Mitchell, A. N. Netravali // SIGGRAPH Comput. Graph: электронный ресурс. – 06.1998. – С. 221–228.

2. *Smith, A. R.* A Pixel Is Not A Little Square, A Pixel Is Not A Little Square, A Pixel Is Not A Little Square! (And a Voxel is Not a Little Cube) / A. R. Smith // Technical Memo 6, Microsoft Research. – 1995.

3. *Beets, K.* Super-sampling Anti-aliasing Analyzed / K. Beets, D. Barron // Beyond3D – 2000.

4. *Dominé, S.* OpenGL Multisample OpenGL Multisample / S. Dominé // Nvidia – 2002.

5. 3D Center, Multisampling Anti-Aliasing: A Closeup View – Режим доступа: http: //alt.3dcenter.org/artikel/multisampling_anti-aliasing/index3_e.php. – (Дата обращения: 14.04.2020).

6. *Nguyen, H.* GpuGems3 / H. Nguyen // Addison-WesleyProfessional – 2007.

7. What is Lanczos resampling useful for in a spatial context? – Режим доступа: https: //gis.stackexchange.com/questions/10931/what-is-lanczos-resamplinguseful-for-in-a-spatial-context. – (Дата обращения: 16.04.2020).

8. OpenGL Programming Guide: The Official Guide to Learning OpenGL / D. Shreiner [и др.] // Addison-Wesley Professional. – 2013.

9. *Ku XU, N. D.* Temporal Antialiasing In Uncharted 4 / N. D. Ku XU // Naughty Dog – 2016.

10. OpenGL ARB texture multisample – Режим доступа: https://www.khronos.org/registry/OpenGL/extensions/ARB/ARB_texture_ multisample.txt. – (Дата обращения: 14.04.2020).

11. *Karis, B. U.* High Quality Temporal Supersampling / U. B. Karis // SIGGRAPH Comput. Graph: электронный ресурс. – 2014.

12. NVIDIAPostWorks – Режим доступа: https://developer.nvidia.com/postworks. – (Дата обращения: 10.04.2020).

Псарев Никита Андреевич – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: nikitapsarev@gmail.com

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 519.876.5

МОДЕЛИРОВАНИЕ ЭПИДЕМИИ ОТДЕЛЬНОЙ ПОПУЛЯЦИИ С НЕКОТОРЫМИ ВЕРОЯТНОСТНЫМИ ХАРАКТЕРИСТИКАМИ

И. Б. Рахимов

Воронежский государственный университет

Введение

Целью статьи является создание модели распространения вируса с некоторыми вероятностными параметрами и оценка их влияния на модель.

1. Модели

Создадим простейшую модель: есть популяция из 228 особей на изолированной территории площади примерно равной 4-м квадратным километрам. Будем считать, что это город. Число особей взято из средней плотности населения Земли на квадратный километр в начале 2020 года – 57 человек на квадратный километр. Есть некий вирус, который появляется случайно у одной из особей и передаётся с некоторой заданной вероятностью при непосредственном контакте заражённой и не заражённой особями [3]. Так же в модели есть дополнительная территория – больница, в которую направляются заражённые особи, если введён карантин. Если карантин не введён, особи передвигаются свободно по всему городу и больнице.

Рассмотрим несколько таких моделей.

Для создания моделей была использована среда имитационного моделирования AnyLogic [1, 2].

1.1. Больница

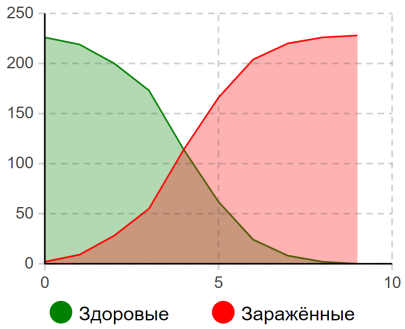


Рис. 1. беспрепятственное распространение вируса

Начнём с простейшей модели: особи передвигаются случайно и заражаются с вероятностью 0.8 при непосредственном контакте. Посмотрим, как быстро все особи будут заражены если не предпринимать никаких действий (рис. 1). Здесь ось Y – количество заражённых/здоровых, ось X – время в единицах модельного времени.

Теперь добавим больницу. Для начала незараженные особи могут зайти в больницу, и могут свободно выйти, но при этом могут быть заражены в ней даже без контакта с заражённым с вероятностью 0.65. Заражённые особи будут госпитализированы в больницу с вероятностью 0.8. Они не могут покидать больницу, так же они могут самостоятельно прийти в больницу, но не смогут выйти. Больница начнёт принимать заражённых после того как их будет больше, чем четверть популяции. Так же после открытия больницы у особей страбатывает инстинкт самосохранения, который заставляет их передвигаться медленее для уменьшения количества встреч.

Посмотрим, с какой скорость распространится вирус в этот раз (рис. 2). Можно заметить, что после открытия больницы темпы заражения снижаются.

Изменим модель следующим образом: теперь здоровые особи могу войти в больницу только через определённую область, но выйти из неё могут откуда угодно. Скорость распространения показана на рис. 3. Заметим, что при таком режиме скорость распространения в начале модели похожа на скорость предыдущих моделей. Однако, рассматривая модель на более длительном промежутке времени, можно увидеть, что распространение вируса после открытия больницы практически останавливается, чего нельзя сказать о модели с открытой больницей.

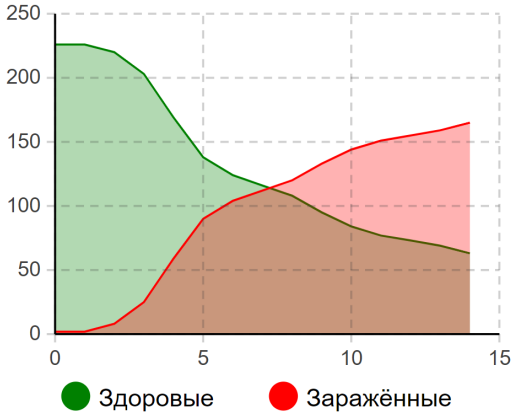


Рис. 2. Открытая больница

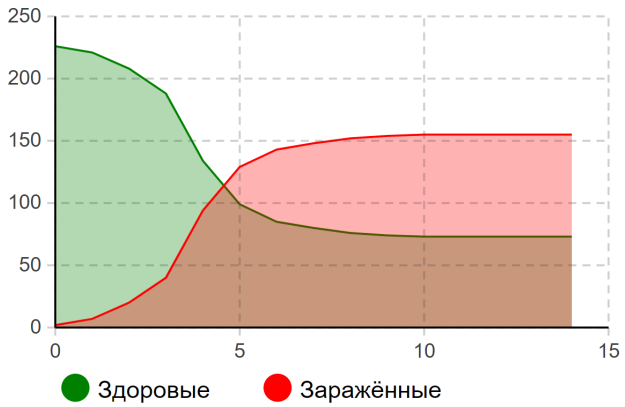


Рис. 3. Закрытая больница

1.2. Выздоровление, смертность и иммунитет

Усложним модель, добавив смерть от болезни спустя 5 единиц модельного времени после заражения с вероятностью 0.4, и обратное событие – выздоровление с вероятностью 1 – 0.4 = 0.6. После выздоровления повторное заражение произойти не может, так как вырабатывается иммунитет.

Нахождение заражённой особи в больнице повышает вероятность её выздоровления на 0.1.

Скорость распространения вируса, количество погибших и иммунных особей в такой модели с больницей и без показано на рис. 4 и рис. 5 соответственно.

Можно заметить, что больница работает достаточно эффективно, часть особей даже не была заражена вирусом, а количество смертей понизилось примерно в два раза.

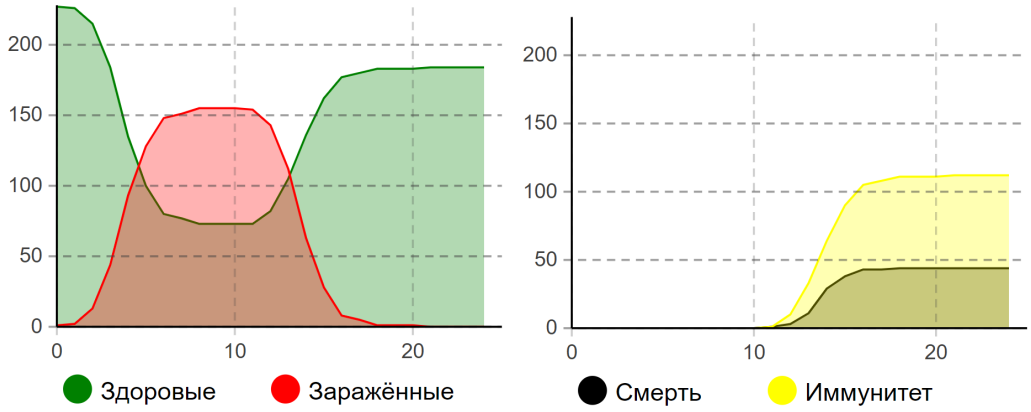


Рис. 4. Больница работает

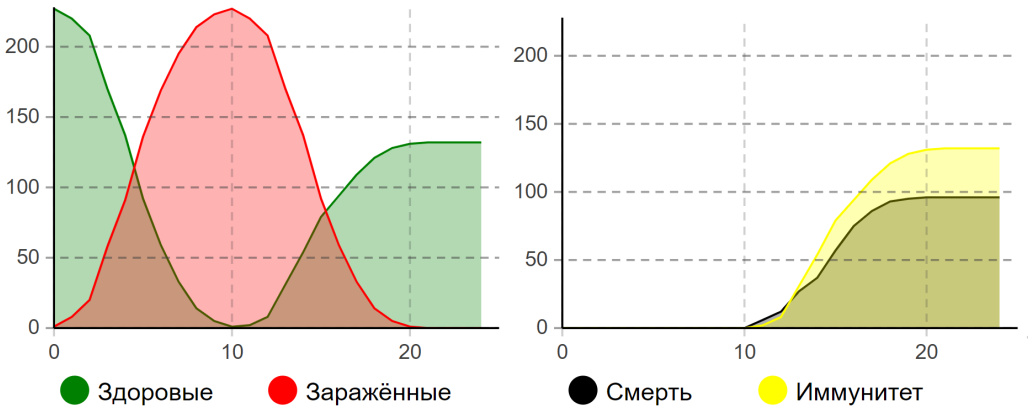


Рис. 5. Больница не работает

2. Вероятность

Исследуем, как все модельные вероятности в последней модели влияют на темпы распространения вируса. А именно, как вероятность заражения при непосредственном контакте и вероятность быть госпитализированным влияют на время существования вируса. Для этого построим трёхмерный график (рис. 6). Зададим минимальную вероятность равную 0.01, а максимальную 0.9. Желтые области соответствуют минимальному времени, бордовые – максимальному.

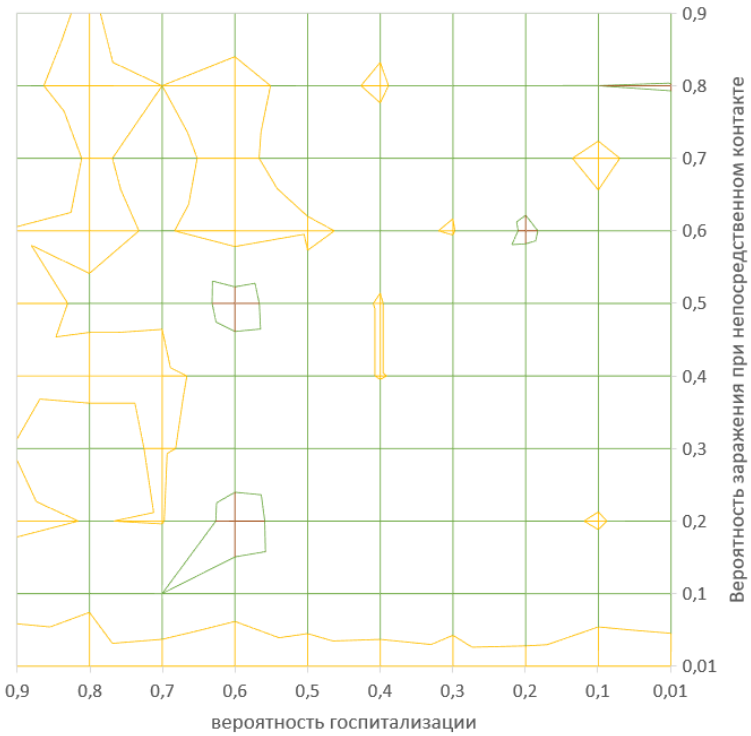


Рис. 6. Зависимость времени существования вируса от вероятности заражения при контакте и вероятность попадания в госпиталь

Заметим, что если вероятность заражения меньше чем 0.1, то вирус исчезает очень быстро, так как просто не способен передаться большому числу особей.

Если вероятность попадания в больницу меньше чем 0.5, то получается средняя длительность протекания эпидемии.

Самое интересное возникает при вероятности попадания в больницу большей чем 0.5. В этой области время эпидемии в основном минимально. При вероятности заражения при непосредственном контакте меньшей чем 0.5, в этой области время существования вируса минимально, потому что больница успевает изолировать всех больных. Но если эта вероятность больше чем 0.5, то минимальное время достигается из-за больших темпов распространения вируса, то есть большая часть популяции будет заражена вирусом сразу. Так же из-за работы больницы появятся особи, которые не были заражены, так как из-за достаточно высокой вероятности попадания в больницу даже большое количество больных особей будет быстро изолированно.

3. Время начала работы больницы

Вернёмся к модели, которая была получена в разделе 2 и изменим время открытия больницы. Пусть теперь она открывается после определенного времени и не привязана к количеству зараженных. Выясним как время открытия больницы влияет на смертность. Для этого построим двухмерный график (рис. 7). Максимальное время на графике соответствует открытию больницы после эпидемии.

Рис. 7. Зависимость смертности от времени открытия больницы

Можно заметить, что после 5 единиц модельного времени открытие больницы почти никак не влияет на смертность, это связано с тем, что за это время все особи уже будут заражены и помещение больных в больницу не так сильно повышает вероятность выздоровления, всего на 0.1.

Заключение

В заключении выделим полученные результаты.

1. Изоляции заражённых положительно влияет на уменьшение темпов эпидемии.
2. Большая вероятность госпитализации способствует быстрому окончанию эпидемии.
3. Раннее открытие больниц даже с плохим уровнем медицины способствует снижению смертности.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Медведев, С. Н.* Имитационное моделирование в среде AnyLogic / С. Н. Медведев, О. А. Медведева, О. Г. Корольков. – Воронеж : ФГБОУ ВО “ВГУ”, 2018. – 74 с.

2. AnyLogic : имитационное моделирование для бизнеса – Режим доступа: https://www.anylogic.ru – (Дата обращения: 24.04.2020).

3. *Keeling, Matt; Rohani, Pej.* Modeling Infectious Diseases: In Humans and Animals. Princeton: Princeton University Press.

Рахимов Ихтиёржон Бахтиёржонович – студент 2-го курса направления Прикладная математика и информатика Воронежского государственного университета. E-mail: ihtier_@outlook.com

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 004.92

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ ДЛЯ ПОСТРОЕНИЯ ТРЕХМЕРНЫХ ПОВЕРХНОСТЕЙ

К. Г. Резников

Воронежский государственный университет

Введение

Трехмерное моделирование играет важную роль в жизни современного общества. Оно позволяет создать прототип будущего сооружения или коммерческого продукта в объемном формате. Сегодня оно широко используется в сфере маркетинга, архитектурного дизайна и развлечений, не говоря уже о науке и промышленности. Например, в науке – визуализация механических процессов, в промышленности – проектирование деталей автомобилей, в сфере развлечений – видеоигры.

1. Программное обеспечение

Существует множество различного программного обеспечения для построения трехмерных моделей. С помощью некоторых программ можно создавать очень реалистичные модели, которые сложно отличить от фотографии, но есть и ряд значительных недостатков:

- закрытый исходный код;
- дорогостоящая лицензия;
- зависимость от операционной системы и вида устройства;
- отсутствие различных способов построения поверхностей и другие.

Существуют программы, которые исключают один или несколько недостатков. Одной из таковых программ является разработанное веб-приложение. За счет того, что веб-приложение выполняется в браузере, оно исключает один из самых главных недостатков – зависимость от операционной системы и вида устройства. Кроме того, веб-приложение имеет структуру, которая позволяет реализовывать различные способы построения моделей. Например, построение кинематических моделей (рис. 1).

2. Кинематические модели

Кинематические модели формируются непрерывным движением в пространстве некоторой линии (образующая или форма) по определенной траектории (направляющая или путь) [2].

Параметрическое уравнение кинематической модели выглядит следующим образом:

$$p(t,\tau) = p_{\text{в}}(\tau)A(t,\tau) + p\ (t),\ t \in [t_0, t_n],\ \tau \in [\tau_0, \tau_m], \qquad (1)$$

где

- $p_o(\tau)$ – образующая линия с центром в начале координат;
- $p_н(t)$ – направляющая линия, вдоль которой переносится центр образующей;
- $A(t,\tau)$ – матрица преобразования;
- $[t_0, t_n]$, $[\tau_0, \tau_m]$ – интервалы изменения параметров.

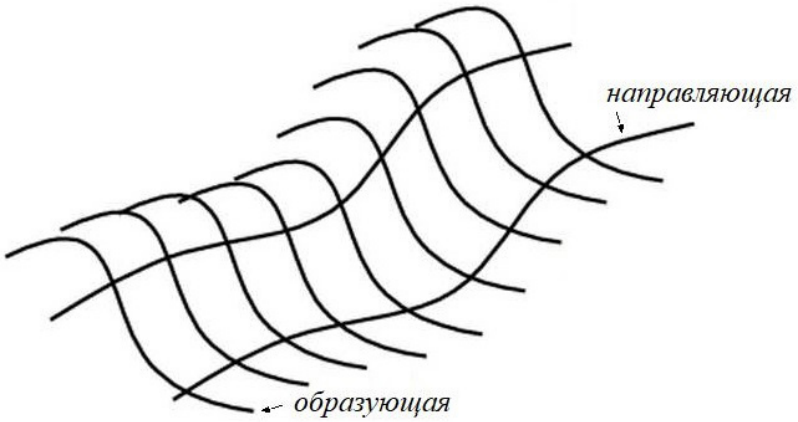


Рис. 1. Кинематическая модель

При задании $p_o(\tau)$ и $p_н(t)$ произвольными функциями возникают некоторые трудности при реализации или вычислениях. Например, чтобы задать кинематическую поверхность сложной формы необходимо подобрать специальные функции или композицию функций. Одним из решений такой проблемы может являться применение методов приближения или интерполяции. Например, $p_o(\tau)$ и $p_н(t)$ можно задать точками, по которым можно вычислить кубический сплайн или интерполяционный полином в форме Ньютона [1].

3. Веб-приложение

Веб-приложение для построения и визуализации трехмерных поверхностей, написано на языке программирования JavaScript и может быть выполнено в любом современном браузере, как персонального компьютера, так и смартфона.

JavaScript поддерживает объектно-ориентированный, императивный и функциональный стиль программирования. JavaScript один из самых популярных, известных и востребованных языков в сфере веб-разработки и программирования в целом [3]. Он имеет огромное количество различных библиотек и фреймворков.

Веб-приложение основано на фреймворке Vue.js.

Vue.js – это прогрессивный фреймворк для создания пользовательских интерфейсов. В отличии от «монолитных» фреймворков, Vue можно внедрять в проект постепенно.

Данный фреймворк помогает создать многофункциональное приложение и дает возможность постоянно расширять и поддерживать проект.

4. Функционал приложения

Прежде всего, приложение предназначено для построения и визуализации кинематических моделей, но функционал приложения значительно шире.

Приложение состоит из следующих частей:

- 2D-сцена для визуализации плоских объектов, таких как наборы точек на плоскости, функции, полиномы, кубические сплайны и другие;
- 3D-сцена для визуализации объемных объектов, таких как кинематические поверхности, каркасные модели и другие;
- модули для работы с математическими моделями и объектами;
- пользовательский интерфейс для взаимодействия с камерой и объектами.

Структура проекта устроена таким образом, чтобы была возможность легко интегрировать дополнительные модули. Например, добавление собственных методов интерполяции или

моделей с различными способами построения. Интерфейс приложения также легко расширяется, как и другие модули.

Рассмотрим функционал основных модулей на примере построения кинематической модели.

5. Построение модели

Итак, открыто главное окно приложения (рис .2).

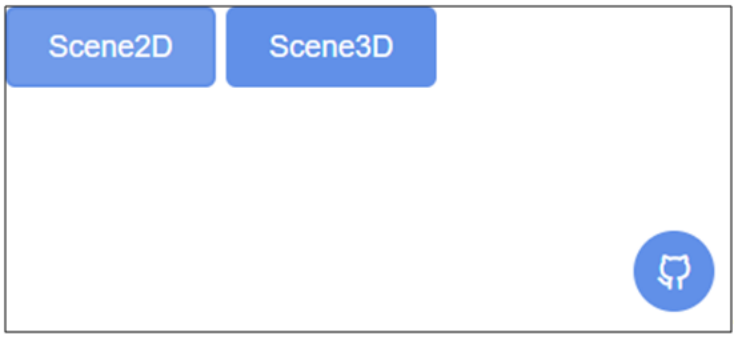


Рис. 2. Главное окно приложения

Шаг 1. Задание образующей и направляющей.

Так как образующая и направляющая это плоские объекты, то необходимо перейти в соответствующую сцену – 2D-сцену (рис. 3).

Задавать образующую и направляющую можно несколькими способами:

- заранее определенной функцией (некоторые математические функции определены по умолчанию);
- набором точек (точки можно задавать 4 способами):
  - ввод координат;
  - клик по плоскости;
  - зажатием левой клавиши мыши и переносом курсора с заданием минимального шага;
  - получение точек по заданной функции;
- полиномом, полученным по заданному набору точек.

Для задания самой кинематической модели следует перейти в 3D-сцену, заранее сохранив созданные объекты для образующей и направляющей в корневом компоненте приложения, так как компоненты 2D-сцены и 3D-сцены не связаны между собой.

Шаг 2. Задание кинематической модели.

Перейдя в 3D-сцену (рис. 4), необходимо создать объект кинематической модели и выбрать в выпадающих списках соответствующие объекты для образующей и направляющей. По умолчанию матрицей преобразования является единичная матрица. В соответствующем выпадающем списке можно выбрать основные матрицы преобразования:

- перенос;
- поворот относительно оси X [Y, Z];
- собственная.

У пользователя есть возможность изменить количество точек на построение образующий и направляющей. По умолчанию, количество точек равно десяти.

После того как все поля заполнены, необходимо сохранить кинематическую модель, с помощью соответствующей кнопки рядом с полями.

Далее модель можно отобразить, а также изменить в пространстве положение образующей и направляющей, переместить поверхность в пространстве с помощью соответствующих клавиш.

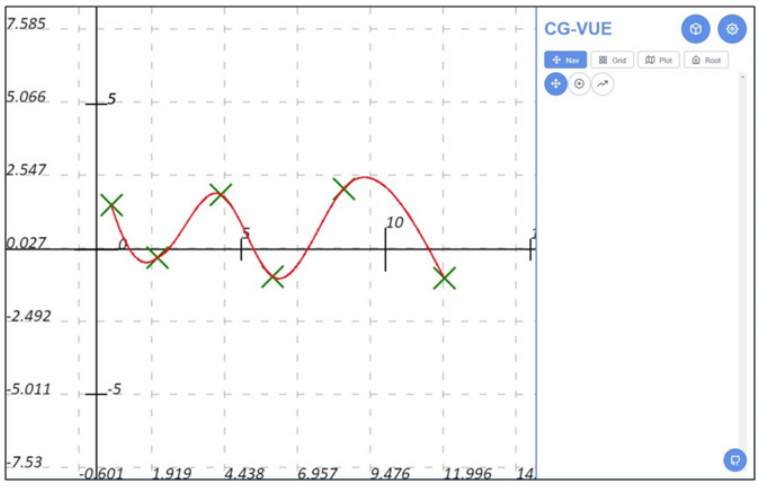


Рис. 3. Скриншот приложения с 2D-сценой

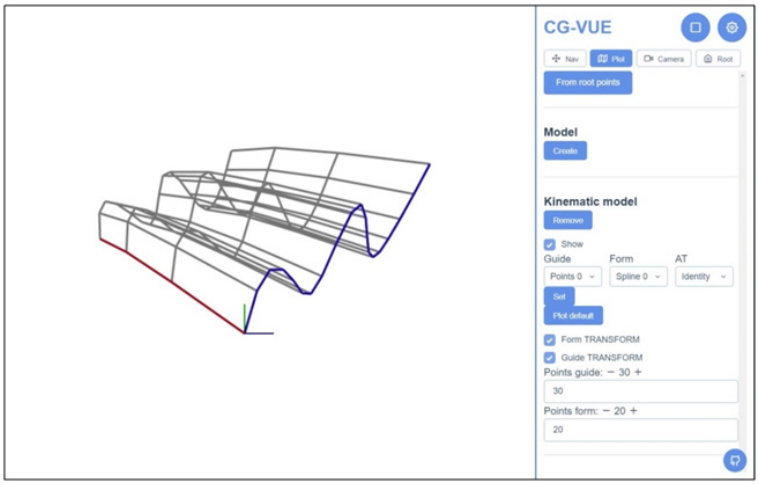


Рис. 4. Скриншот приложения с 3D-сценой

Заключение

Воспользоваться приложением может любой желающий по адресу – http://reznikovk.ru/nm-vue.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Бахвалов, Н. С.* Численные методы: учебное пособие / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. – Санкт-Петербург : Физмалит, 2002. – 630 с.

2. *Никулин, Е. А.* Компьютерная геометрия и алгоритмы: учебное пособие / Е. А. Никулин. – Санкт-Петербург : БХВ-Петербург, 2003. – 560 с.

3. Stack Overflow Developer Survey 2018. – Режим доступа: https://insights.stackoverflow.com/survey/2018#most-popular-technologies

Резников Константин Георгиевич – магистрант 1-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: rkg@reznikovk.ru

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц. кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 004.021

ИССЛЕДОВАНИЕ СПОСОБОВ РАСПОЗНАВАНИЯ СИМВОЛОВ НА БАНКОВСКОЙ КАРТЕ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ МАШИННОГО ОБУЧЕНИЯ

Е. Д. Свиридова, С. Ю. Болотова

Воронежский государственный университет

Введение

В современном мире ежедневно онлайн совершается огромное число платёжных операций. При этом возникает необходимость постоянного ручного ввода данных банковских карт.

Данная работа посвящена созданию мобильного приложения, позволяющего в режиме реального времени распознать данные банковской карты при помощи камеры мобильного устройства. Разработанная программа находит карту в видеопотоке, распознает на ней цифровые и буквенные символы, при этом позволяя извлекать данные даже в условиях плохой освещенности и неоднородности фонового изображения.

1. Постановка целей и задач

Целью работы является исследование различных способов и методов по распознаванию текста с изображения с применением технологии машинного обучения. Такое исследование поможет найти оптимальное решение, которое бы выдавало наиболее точные результаты в процессе распознавания.

В соответствии с указанной целью требуется решить следующие задачи:

1. Анализ существующих способов распознавания текстовой информации из видеопотока.

2. Разработка решения, которое сочетает в себе достоинства существующих подходов и исключает недостатки, связанные со сложностью реализации и интеграции, невысокой скоростью распознавания.

В рамках работы применение такого решения показано на примере распознавания текстовой информации, расположенной в прямоугольной области на изображении, получаемом с камеры мобильного устройства. Такой областью является лицевая сторона банковской карты, которая представляет собой документ стандартных размеров (определены стандартом ISO/IEC 7810) с определенным набором полей, а именно: номер карты, имя и фамилия владельца карты, срок действия. Их расположение может различаться в зависимости от банка, выпустившего карту, также могут присутствовать и другие необязательные поля. С точки зрения распознавания, процесс осложняется наличием пестрого и неоднородного фона банковской карты, условиями освещения, при которых было получено изображение, наличия дефектов в виде потёртостей на старых картах.

В силу того, что распознавание происходит на мобильном устройстве (в рамках работы в качестве мобильной платформы используется платформа iOS), требуется решить ряд вопросов, которые относятся к процессу подготовки данных и постобработки результатов с целью выявления ошибок распознавания. Поэтому весь процесс реализации можно на разбить на следующие этапы:

– локализация прямоугольной области банковской области в видеопотоке;

– применение к полученному прямоугольному изображению проективного преобразования с целью коррекции перспективы изображения и приведения его к фиксированному разрешению;

– применение фильтров к изображению для подавления фонового изображения банковской карты;

– поиск информационных полей (строк);

– сегментация найденных строк на отдельные символы;

– распознавание символов с помощью алгоритмов машинного обучения;

– обработка результатов распознавания для повышения точности полученных данных.

Таким образом, выполнение данной работы заключается в исследовании вариантов решения рассмотренных выше задач, проведении сравнительного анализа полученных данных, проведении экспериментов, задающих наиболее сложные условия для распознавания, и выборе того решения, что выдаёт наиболее достоверные результаты.

2. Реализация

Первым этапом в процессе реализации является определение в видеопотоке прямоугольной области, которое может осуществляться несколькими способами, а именно:

1. Использование средств компьютерного зрения для определения прямоугольной области в видеопотоке, а именно стандартного для мобильной платформы iOS фреймворка Vision.

2. Выделение пользователем границ карты вручную.

Второй способ демонстрирует наиболее высокую точность в определении прямоугольной области в сравнении с первым, поскольку исключает вероятность возникновения ошибок, связанных с нечетким отображением границ карты. Однако его главным недостатком является запрос на выполнение пользователем дополнительных действий, что может негативно повлиять на его общее впечатление от приложения. Поэтому в работе используется первый способ, демонстрация результата его применения в работе представлена на рис. 1.

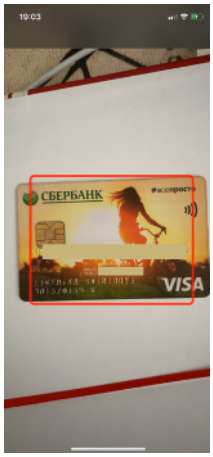


Рис. 1. Демонстрация результата определения прямоугольной области в видеопотоке

Затем осуществляется преобразование нормализованных координат углов полученной прямоугольной области в экранные с последующим применением к ним фильтра коррекции перспективы. Таким образом, получаем преобразованное прямоугольное выходное изображение, которое поступает на вход последующим этапам.

Далее на изображении требуется определить информационные поля и «разбить» получившиеся строки на отдельные символы. На данном этапе эта часть ещё не реализована, её полное описание с полученными результатами будет отражено в основной работе, которой и посвящена данная публикация. Однако для решения данной задачи исследуются результаты применения такого метода, как алгоритм сегментации, основанный на утверждении, что средняя яркость изображений межстрочных интервалов существенно ниже средней яркости в изображении текстовых строк. Другим вариантом является применение методов машинного обучения. Такой подход обладает большим преимуществом, так как для него не требуется предварительная обработка и постобработка данных. Важность декомпозиции изображения строки на фрагменты изображений обусловлена тем, что в рамках работы стоит задача распознавания отдельных символов, а не целых фрагментов текста. Некорректность проставления разрезов между символами может явиться причиной значительной доли ошибок конечного распознавания.

Предполагается, что вне зависимости от выбранного решения на выходе получим изображение отдельного символа. На рис. 2 представлены примеры изображений для символа «2».

Рис. 2. Пример изображения символа «2»

Такое изображение поступает на вход модели, которая интегрируется в мобильное приложение посредством фреймворка CoreML. Модель распознавания символов по их изображению состоит из 4-х скрытых сверточных слоёв и одного выходного плотного слоя. В качестве оптимизатора был выбран стохастический градиентный спуск (SGD) с категориальной кросс-энтропией как функцией потери. Следует отметить, что точность распознавания зависит от качества и размера обучающей выборки.

В связи с тем, что банковская карта является документом, содержащим персональные данные человека, довольно сложно сформировать выборку размера, достаточного для обучения. Поэтому в рамках работы потребовалось искусственно увеличить размер выборки с помощью преобразования уже имеющихся данных. Также за счёт такого расширения набора данных можно выровнять количество изображений для каждого символа, чтобы не возникло ситуации, когда количество изображений того или иного символа преобладает.

Ввиду того, что искусственная нейронная сеть может ошибаться и выдавать неточные результаты распознавания, необходимо применение методов постобработки. Так как в рамках работы производится распознавание символов банковской карты, то для выявления ошибок распознавания цифр карты используется алгоритм Луна, наложения определенных ограничений на полученный срок действия карты также поможет устранить ошибки распознавания (к примеру, не может быть 15-го месяца), имя владельца карты должно исключать наличие цифр.

Заключение

Результатом исследования является узконаправленное мобильное приложение на платформе iOS, которое при условии сочетания всех реализованных решений демонстрирует высокую точность распознавания символов на банковской карте в видеопотоке. На данном этапе точность распознавания достигает ~0.97 % на обучающей выборке и 95 % на тестовой.

Литература

1. ГОСТ Р ИСО/МЭК 7810-2015. Карты идентификационные. Физические характеристики. – Введ. 2017-01-01. – Москва : Стандартинформ, 2018. – 22 с.

2. *Петкун, А. В.* Нейронная сеть как средство для распознавания пиксельных изображений цифр банковских карт / А. В. Петкун, Е. В. Романов, А. В. Фисунов // Научное сообщество студентов XXI столетия. Технические науки: сб. ст. по мат. LXI междунар. студ. науч.-практ. конф. – Новосибирск: Изд. АНС «СибАК», 2018. – № 1 (60). – С. 262.

3. Легко ли распознать информацию на банковской карте. – Режим доступа: https://habrahabr.ru (дата обращения: 18.04.2020).

4. *Джулли, А.* Библиотека Keras – инструмент глубокого обучения. Реализация нейронных сетей с помощью библиотек Theano и Tensorflow / А. Джулли, С. Пал. – Москва : ДМК Пресс, 2018. – 294 с.

5. Документация к фреймворку Core ML. – Режим доступа: https://developer.apple.com/documentation/coreml (дата обращения: 17.04.2020).

6. Документация к фреймворку Vision. – Режим доступа: https://developer.apple.com/documentation/vision (дата обращения: 17.04.2020).

Свиридова Евгения Дмитриевна – магистрант 2-го года обучения кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: sdevgenia@gmail.com

Болотова Светлана Юрьевна (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: bolotova.svetlana@gmail.com

УДК 004.78

АВТОМАТИЗАЦИЯ ОТВЕТОВ НА ЧАСТО ЗАДАВАЕМЫЕ ВОПРОСЫ ПРИ ПОМОЩИ ГОЛОСОВОГО ЧАТ-БОТА

Д. В. Симонов

Воронежский государственный университет

Введение

Страницы часто задаваемых вопросов – это самый распространенный способ для организаций уменьшить нагрузку на службу поддержки пользователей, автоматически отвечая на наиболее часто задаваемые вопросы. Однако с течением времени количество вопросов может оказаться слишком большим и поиск нужного будет затруднительным для пользователя. В последние годы стали значительно популярны голосовые чат-боты. С их помощью пользователи могут узнать время, погоду или новости. В то время как предыдущее поколение чат-ботов было основано на правилах «если-то», сегодняшние чат-боты используют машинное обучение и искусственный интеллект для имитации диалогов с пользователями на естественном языке. В данной работе будет представлено приложение для ответов на часто задаваемые вопросы о факультете ПММ с использованием голосового чат-бота. Для поиска близкого к заданному вопроса из готового набора пар «вопрос-ответ» была использована модель «мешок слов». Серверная часть приложения была написана на python с помощью библиотеки flask. Клиентская часть реализована в виде мобильного приложения на операционной системе Android.

1. Сбор данных

Для формирования набора пар «вопрос-ответ» использовались данные, собранные с официального сайта ПММ ВГУ. Большая часть данных получена путем парсинга раздела «Вопрос декану». Для этого был написан парсер на python с использованием библиотеки Beautiful Soup 4. Также некоторые данные такие, как должности и контакты для связи, были внесены вручную. Все пары занесены в отдельный csv файл, который используется в дальнейшем в работе модели.

2. Модель

Для определения семантической близости предложений при построении диалогов в моей работе была выбрана модель «мешок слов» с модификацией tf-idf. Создание представления текста в виде мешка слов происходит в 3 этапа:

1. *Токенизация* (*tokenization*). Разбиение каждого документа на слова, которые встречаются в нём с помощью пробелов и знаков пунктуации.

2. *Построение словаря* (*vocabulary building*). Словарь собирается из всех слов, которые появляются в любом из документов, после чего каждое слово пронумеровывается (например, в алфавитном порядке).

3. *Создание разреженной матрицы* (*sparse matrix encoding*). Для каждого документа подсчитывается частота появления в документе каждого слова из словаря.

Для каждого слова в документе указывается некоторый «вес», который вычисляется по следующей формуле [1]:

$$tfidf\left(t,d,D\right)=tf\left(t,d\right)\cdot idf\left(t,D\right).$$

Здесь *TF* (*term frequency*) – это отношение числа вхождений некоторого слова к общему числу слов в документе, которое вычисляется следующим образом:

$$tf(t,d) = \frac{n_t}{\sum_k n_k},$$

где n_t это число вхождений слова t в документ, $\sum_k n_k$ – общее число слов в данном документе.

IDF (*inverse document frequency*) – это инверсия частоты, с которой некоторое слово встречается в документах корпуса:

$$idf(t,D) = \log \frac{|D|}{\left|\left\{d_i \in D \middle| t \in d_i\right\}\right|},$$

где $|D|$ – число документов в корпусе, а $\left|\left\{d_i \in D \middle| t \in d_i\right\}\right|$ – число документов из корпуса D, в которых встречается t.

Для вычисления сходства векторов используется коэффициент Отиаи (cosine similarity) [2], равный косинусу угла между векторами:

$$sim_{cos}(u,v) = \frac{u \cdot v}{\|u\| \cdot \|v\|}.$$

В библиотеке deeppavlov [3] для python доступна предобученная модель, где в роли корпуса текстов используется русскоязычный раздел википедии. Принцип работы модели следующий:

1. На вход модели подаётся текст.
2. Создается представление текста в виде вектора.
3. Вычисляется сходство полученного вектора со всеми векторами вопросов из набора пар «вопрос-ответ».
4. Выбирается вектор с наибольшим коэффициентом.
5. Если коэффициент сходства больше порогового (установленного пользователем), возвращается ответ из этой пары, иначе возвращается стандартный ответ для отсутствия сходства.

Для представления диалога был реализован класс AmmChatBot, содержащий следующие методы:

1. init – инициализация модели и загрузка набора «вопрос-ответ» из файла.
2. ask – метод принимающий вопрос и возвращающий ответ.
3. get_dialog – метод, позволяющий получить заданные ранее вопросы и полученные на них ответы в виде списка.

3. Архитектура приложения

Клиент-серверное приложение состоит из 2 частей: мобильное приложение на операционной системе Android и веб-сервер, написанный на python. Схема его работы изображена на рис. 1.

Веб-сервер написан с использованием библиотеки flask. Его задача состоит в том, чтобы обрабатывать post запрос в формате JSON от клиента содержащий текст вопроса, передавать его модели и полученный ответ отсылать клиенту.

Android приложение написано на языке java с использованием библиотеки retrofit2 для обеспечения сетевого взаимодействия. Функции преобразования речи в текст (speech recognition) и синтеза речи (text-to-speech) реализованы с помощью встроенной библиотеки android.speech.

Описать работу приложения можно следующим образом:

1. Пользователь нажимает на кнопку записи голоса.

Рис. 1. Схема работы приложения

2. Записанная речь преобразуется в текст.
3. Текст отправляется на сервер в json формате post запросом.
4. Сервер обрабатывает полученное сообщение и отправляет на вход модели.
5. Модель обрабатывает вопрос и выдает ответ.
6. Сервер отправляет полученный ответ android приложению.
7. После получения ответа в виде текста приложение преобразует текст в речь.

Заключение

В результате работы был разработан голосовой чат-бот, отвечающий на вопросы о ПММ. Базу вопросов-ответов можно обновлять и пополнять и тем самым увеличивать и актуализировать информацию.

Литература

1. TF-IDF. – Режим доступа: https://ru.wikipedia.org/wiki/TF-IDF. (Дата обращения: 2.05.2019)
2. Cosine similarity. – Режим доступа: https://en.wikipedia.org/wiki/Cosine_similarity. (Дата обращения: 9.05.2019)
3. DeepPavlov's documentation. – Режим доступа: http://docs.deeppavlov.ai/en/master/. (Дата обращения 19.05.2019)

Симонов Дмитрий Валерьевич – магистрант 2-го года обучения кафедры математических методов исследования операций факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: smnv96@yandex.ru

УДК 004.92

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ РАБОТЫ С ЛАБИРИНТАМИ

Э. С. Симонян

Воронежский государственный университет

Введение

Данная статья посвящена разработке ПО, функционал которого будет предоставлять пользователю все необходимые возможности для работы с лабиринтами: построение лабиринта различными способами, нахождение всех путей в нем, а также мини-игры по прохождению лабиринта и прочие дополнительные утилиты. Весь функционал и необходимые утилиты будут описаны в основном тексте статьи.

1. Практическая часть

Для демонстрации реализации некоторых подходов к генерации лабиринтов и их прохождения была разработана программа Maze Builder (Строитель лабиринтов). Языком программирования был выбран C#. Главные функции программы включают в себя:

1. Генерирование лабиринтов при помощи двух встроенных алгоритмов.
2. Рисование лабиринтов пользователем на форме.
3. Поиск всех путей в лабиринте при помощи модификации муравьиного алгоритма и их показ на поле лабиринта.
4. Мини-игры по прохождению лабиринта пользователем в двумерном и трехмерном вариантах.

Окно программы поделено на три части (рис. 1). Левая половина окна отведена под визуализацию лабиринта. Технически визуализация представляет собой таблицу без перегородок, размеры ячеек которой адаптируются под размер генерируемого лабиринта до некоторого предела. Пределом является такой размер ячейки, при котором лабиринт невозможно рассмотреть. В таком случае добавляются ползунки прокрутки поля. Ячейки, являющиеся в лабиринте стенками, закрашиваются черным цветом, точки входа и выхода закрашиваются зеленым и красным цветами соответственно, остальные клетки, по которым по лабиринту можно передвигаться, остаются белыми. Несмотря на то, что метод создания лабиринта в виде bitmap-карты является более экономным с точки зрения потребляемых ресурсов, визуализация таблицей имеет свои преимущества:

1. При наведении мыши на клетку есть доступ как к пиксельным координатам курсора, так и к непосредственно координатам клетки лабиринта. В случае с bitmap-картой координаты клетки лабиринта придется каждый раз пересчитывать из пиксельных координат.
2. Размеры клетки также доступны в любой момент и не требуют пересчета.

Справа в верхней части находится панель с различными настройками и режимами, в правом верхнем углу показываются описание и инструкции по работе с программой (показываются при наведении курсора на объект), снизу расположена консоль для вывода дополнительной информации.

Панель имеет четыре вкладки (рис. 2): первая содержит все инструменты для генерирования и рисования лабиринта, вторая – утилиты для поиска путей, остальные отвечают за

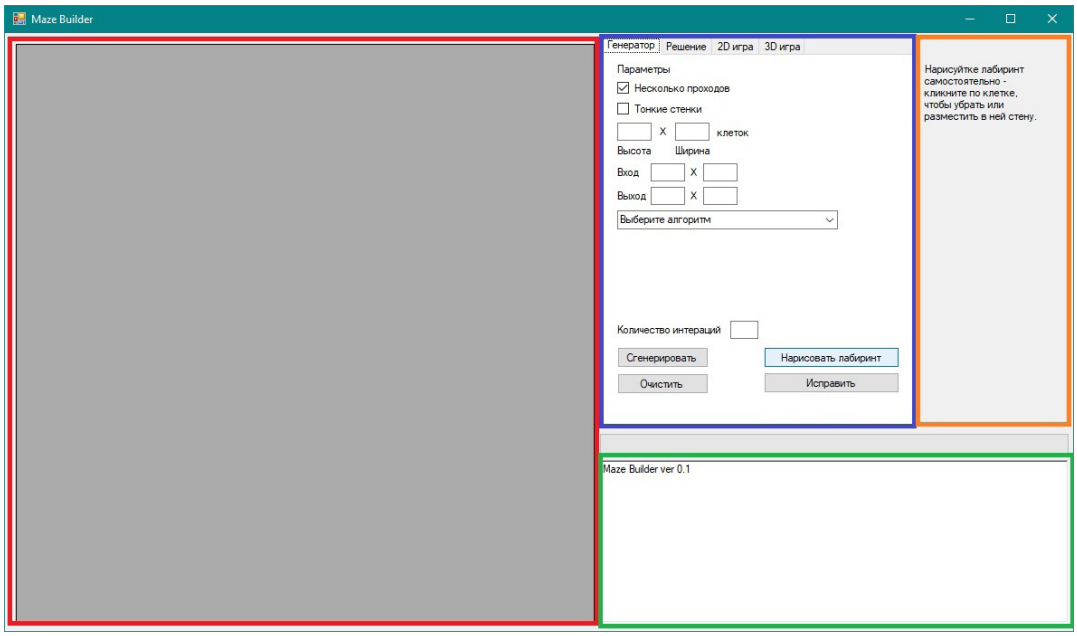


Рис 1. Основные компоненты окна программы

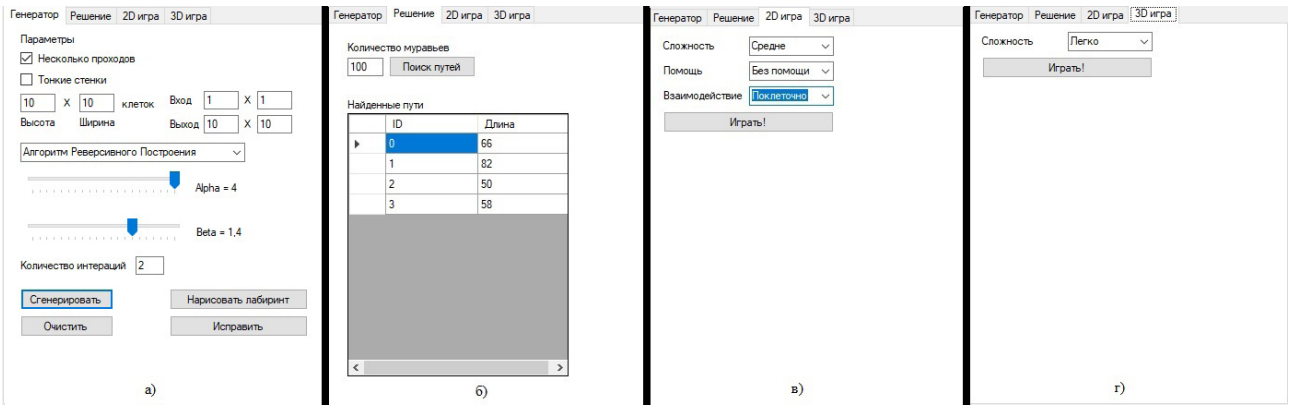


Рис 2. Вкладки программы: а) генератор, б) поиск путей, в) мини-игра 2D, г) мини-игра 3D

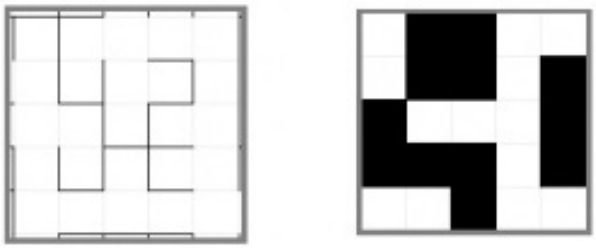


Рис 3. Лабиринт с тонкими стенками (слева) и с толстыми стенками (справа)

настройку и запуск мини-игр по прохождению двумерного и трехмерного лабиринтов соответственно. Рассмотрим каждую вкладку подробнее.

Вкладка «Генератор» содержит выбор возможности генерировать лабиринт с одним или несколькими проходами, выбор визуализации лабиринта «тонкими» или «толстыми» стенками (разница показана на рис. 3), поля ввода высоты и ширины лабиринта, координаты входа и выхода и настроечные параметры для двух алгоритмов генерации.

Также на форме присутствует функция ручного «рисования» лабиринта пользователем: при нажатии кнопки «Нарисовать лабиринт» создается заготовка лабиринта (рис. 4а), при нажатии кнопкой мыши по любой разрешенной клетке в этих координатах убирается или ставится стена. При этом после каждого действия на поле лабиринта запускается алгоритм поис-

ка в ширину, который одновременно проверяет возможность найти путь от входа к выходу и ищет недосягаемые области. Клетки, в которые можно попасть с точки входа, обозначаются белым цветом, оранжевым цветом обозначаются недосягаемые с точки старта области. На форме при этом показывается информация о том, решаем ли лабиринт, и сколько в данный момент на поле недосягаемых клеток. Лабиринт можно исправить как самостоятельно, так и воспользовавшись автоматической достройкой – в этом случае недостающие ветки лабиринта достраивает базовый алгоритм Уилсона. Примеры работы показаны на рис. 4б, в.

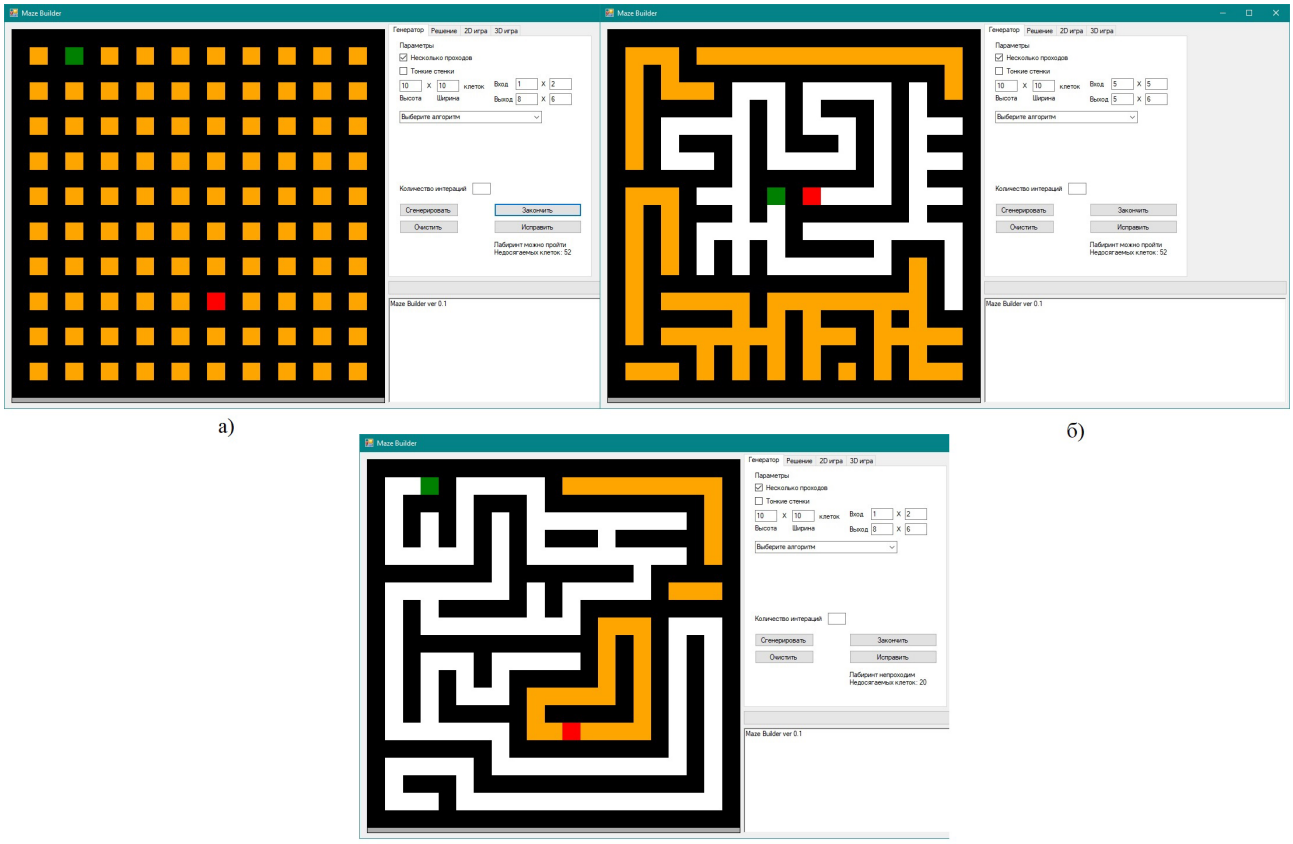
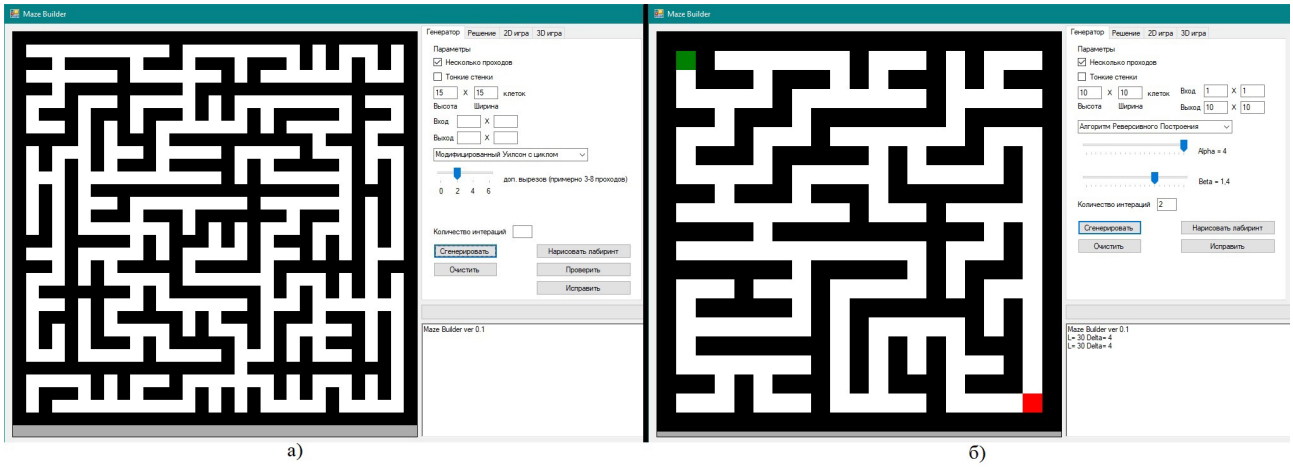


Рис 4. а) Заготовка для рисования лабиринта; б) в) Рисование лабиринта пользователем

Для генерации используются два разработанных алгоритма – модифицированный Уилсон с циклом [1] и алгоритм Реверсивного Построения (рис. 5). Общая идея второго алгоритма состоит в том, что сначала на пустое поле помещаются пути, после чего алгоритмом Уилсона достраиваются тупиковые ветки. Алгоритм старается минимизировать пересечение путей и получить их длины не ниже некоторой заранее вычисленной длины с погрешностью. Оба алгоритма имеют свои настройки.

Вторая вкладка посвящена поиску всех возможных путей в лабиринте (рис. 6). Здесь используется модификация муравьиного алгоритма [1], настройки алгоритма программа предлагает в зависимости от алгоритма и настроек, выбранных при генерации. Также вкладка содержит таблицу с найденными путями и их длинами. Клик на любой номер пути показывает его на самом лабиринте.

Вкладки, отвечающие за запуск мини-игр по сгенерированному или нарисованному лабиринту, содержат настройки игры и кнопку запуска. 2D версия содержит выбор уровня сложности, выбор помощи (активная помощь непосредственно помогает игроку продвинуться, используя подсказки, пассивная же изначально помещает в лабиринт пометки-ориентиры)



а) б)

Рис 5. Лабиринты, построенные: а) Модифицированным алгоритмом с циклом, б) Алгоритмом Обратного Построения

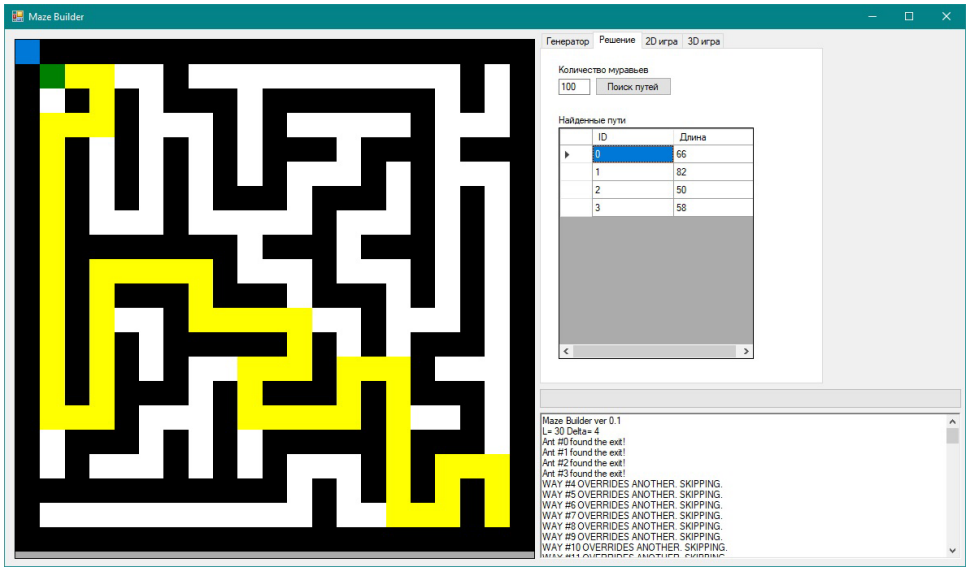


Рис 6. Поиск путей в лабиринте

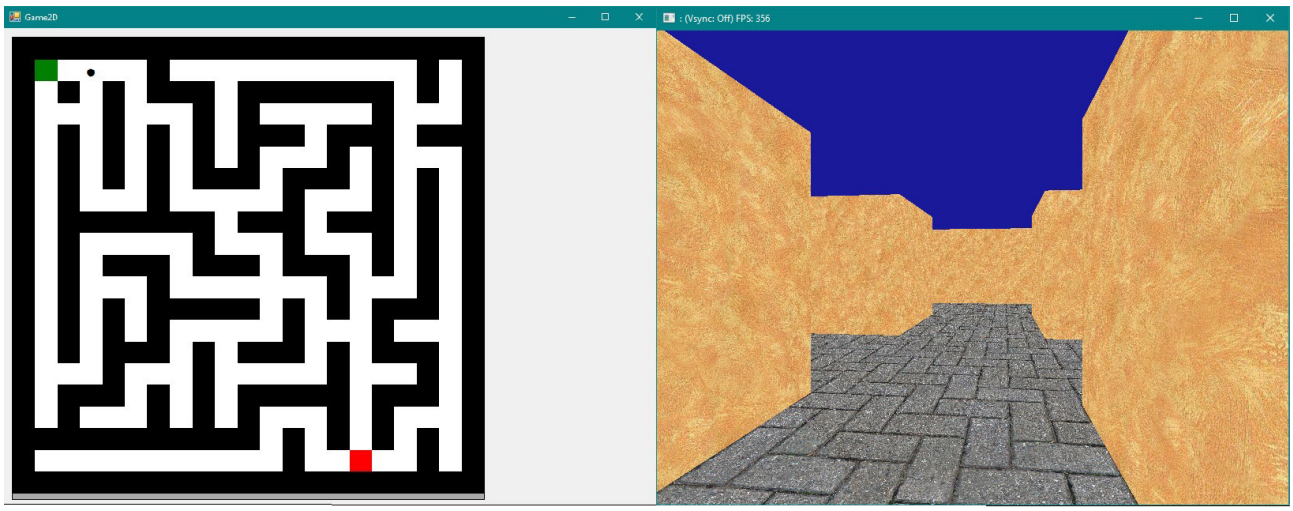


Рис 7. 2D и 3D мини-игры по прохождению лабиринта

и способ перемещения (переход по клеткам или по пикселям). Сама игра запускается в новом окне, управление осуществляется стрелками в 2D варианте. 3D вариант содержит выбор уровня сложности, управление производится кнопками WASD и стрелками. Визуализация мини-игр представлена на рис. 7. Для 3D визуализации выбрана спецификация OpenGL на основе библиотеки OpenTK [2–4].

Заключение

В данном ПО были применены уже существующие алгоритмы, а также разработаны новые. Вся математическая модель была перенесена на более удобные для пользователя формы. Также были реализованы мини-игры с использованием форм и библиотеки OpenTK.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н. и канд. физ.-мат. наук Медведевой О. А.

Литература

1. *Симонян, Э. С.* Создание лабиринта с несколькими проходами, поиск всех путей в нем и их редактирование / Э. С. Симонян, О. А. Медведева, С. Н. Медведев // Моделирование, оптимизация и информационные технологии, Том 7. – 2019.– № 2.– С. 365.

2. Learn OpenTK. – Режим доступа: https://opentk.net/learn/index.html. – (Дата обращения: 02.04.2020).

3. OpenGL 4 with OpenTK in C#. – Режим доступа: http://dreamstatecoding.blogspot.com/p/opengl4-with-opentk-tutorials.html. – (Дата обращения: 08.04.2020).

4. Neo Kabuto's Blog. – Режим доступа: http://neokabuto.blogspot.com. – (Дата обращения: 12.04.2020).

Симонян Эрнест Сергеевич – магистрант 2-го года обучения кафедры Вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: eric96@yandex.ru.

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

Медведева Ольга Александровна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 519.854.2 : 004.4

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ ПОСТРОЕНИЯ ГОРОДСКИХ ЭКСКУРСИОННЫХ МАРШРУТОВ

А. С. Смирнов

Воронежский государственный университет

Введение

В данной статье рассматривается разработка мобильного приложения, демонстрирующего практическое применение муравьиных алгоритмов. В первой части статьи описывается математическая постановка задачи, затем описывается интерфейс приложения, а потом проведен вычислительный эксперимент, демонстрирующий поиск оптимального туристического маршрута и сравнение результатов, полученных жадным и классическим муравьиным алгоритмом.

1. Задача коммивояжера

Задача коммивояжера (Traveling Salesman Problem) заключается в поиске самого оптимального маршрута, который проходит через все вершины графа без повторений с возвратом в исходную вершину.

1.1. Математическая постановка задачи

Математическая постановка задачи может быть записана следующим образом.

Пусть $G = (V, E)$ – связный взвешенный граф, где:

- $V = (v_1, v_2, ..., v_n)$ – конечное множество вершин;
- $E = (e_1, e_2, ..., e_m)$ – конечное множество рёбер;
- d_{ij} – расстояние между вершинами v_i и v_j.

Требуется найти такое подмножество рёбер, которое образует в графе G замкнутый путь, проходит через каждую вершину ровно один раз и обладает минимальной длиной.

Введём переменные x_{ij}, где $x_{ij} = 1$ будет означать, что ребро между вершинами v_i и v_j входит в искомый маршрут. Тогда задача будет заключаться в определении минимума целевой функции:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} x_{ij} \to \min.$$

Существует множество алгоритмов решения этой задачи, среди них:

- полный перебор;
- «жадный» алгоритм;
- метод ветвей и границ;
- генетический алгоритм;
- муравьиный алгоритм.

В данной стать мы рассмотрим более подробно «жадный» и муравьиный алгоритмы.

1.2. «Жадный» алгоритм

Жадный алгоритм – алгоритм нахождения наикратчайшего пути, с помощью выбора на каждом шаге самого короткого, ещё не выбранного ребра. В этом случае решение находится

достаточно быстро, но оно может оказаться не оптимальным. Например, на рис. 1 представлен граф из 4-х вершин с заданными расстояниями между ними. Необходимо обойти все вершины по одному разу и вернуться в исходную. Предположим, что стартовая вершина – вершина с номером 1. В этом примере, жадный алгоритм получит путь 1-2-3-4-1 длиной 22 (3+4+5+10), который не будет кратчайшим за счёт длинного перехода из вершины 4 в вершину 1.

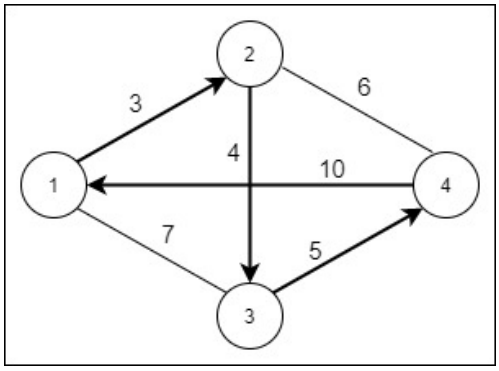


Рис. 1. Граф

1.3. Муравьиный алгоритм

Идея муравьиного алгоритма заключается в имитации поведения муравьёв в природе. Муравьи способны достаточно быстро находить кратчайший путь от гнезда до источника пищи, даже в отсутствии визуального контакта. Это достигается за счет того, что в процессе движения, муравей оставляет на своём пути особый фермент, называемый феромоном. Это вещество воспринимается другими муравьями, и они стараются придерживаться пути с наибольшей концентрацией феромона.

На рис. 2–4 рассматриваются эксперименты с муравьями, показывающие как муравьи находят кратчайшие пути в обход препятствий. Первоначально муравьи обходят препятствие с обеих сторон, но через некоторое время, концентрация феромона на более коротком пути станет больше из-за того, муравьи проходят этот путь быстрее, а значит, что всё больше муравьёв будут выбирать более короткий путь.

Рис. 2. Движение муравьёв без препятствия

Рис. 3. Первоначальное движение муравьёв с препятствием

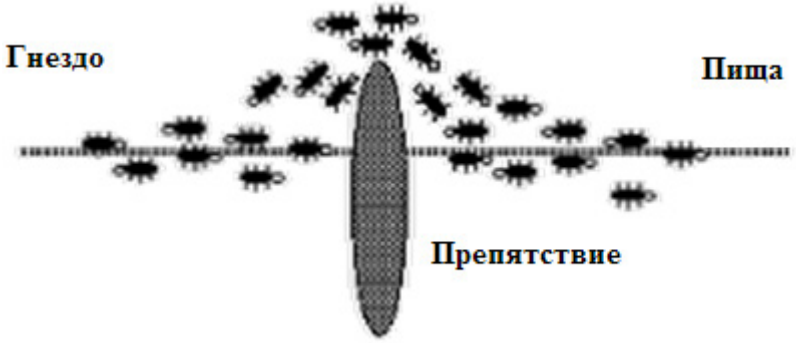


Рис. 4. Выбор муравьями кратчайшего пути в обход препятствия

2. Программная реализация

Мобильное приложение реализовано на платформе Android. Для разработки мобильного приложения использовалась интегрированная среда разработки Android Studio.

2.1. Пользовательский интерфейс

Приложение состоит из трёх основных Activity.

На главном экране приложения располагается фрагмент с картой Google Maps (рис. 5), на котором красными маркерами отмечены достопримечательности города Воронежа, такие как музеи, памятники, парки и другие, популярные среди туристов места. Также, здесь располагается кнопка для перехода к списку достопримечательностей.

На рис. 6 изображено окно, где достопримечательности представлены удобным списком. При нажатии на кнопку в правой части карточки, она «раскроется» и в ней отобразится расширенная информация об этом месте, такая как адрес и рейтинг, рассчитанный на основе оценок пользователей. При нажатии на карточки, пользователь выбирает достопримечательности, которые будут использоваться в решении задачи коммивояжёра по поиску самого выгодного маршрута, проходящего через выбранные точки. В верхней части экрана находятся различные переключатели, позволяющие выбрать начальную и конечную точки маршрута (в данной статье рассматривается только задача коммивояжёра с посещением всех мест только по одному разу с возвратом в исходную точку, поэтому активен переключатель «вернуться в пункт отправления»).

После выбора всех настроек при нажатии на кнопку в правой нижней части экрана, осуществляется переход к третьему Activity, где отображаются несколько наилучших маршрутов с указанием общей длины маршрута, общего времени в пути, а также информация о каждом промежуточном шаге. Примеры результатов поиска кратчайшего маршрута представлен на рис. 7, 8.

2.2. Переход от реальной карты к графу

Математическая постановка задачи коммивояжера требует задания графа, все вершины которого соединены между собой. Так же требуется знать длины ребер между вершинами. Приложение предоставляет список достопримечательностей.

Для получения списка достопримечательностей приложение использует Google Places API. Выбранные достопримечательности являются вершинами графа.

Для расчета длины ребер между вершинами используется время (расстояние) необходимое, чтобы добраться от одной достопримечательности, к другой. Чтобы получить эти данные, для каждой пары вершин используется Google Directions API.

В результате, из полученных вершин и ребер с указанием длины, формируется граф, на котором и решается задача коммивояжера.

3. Проведение вычислительного эксперимента

Целью эксперимента является демонстрация поиска кратчайшего экскурсионного маршрута в мобильном приложении, и сравнение результатов решения задачи, полученных с помощью «жадного» и классического муравьиного алгоритмов.

Вычислительный эксперимент проводился на примере поиска кратчайшего экскурсионного маршрута в городе Воронеж.

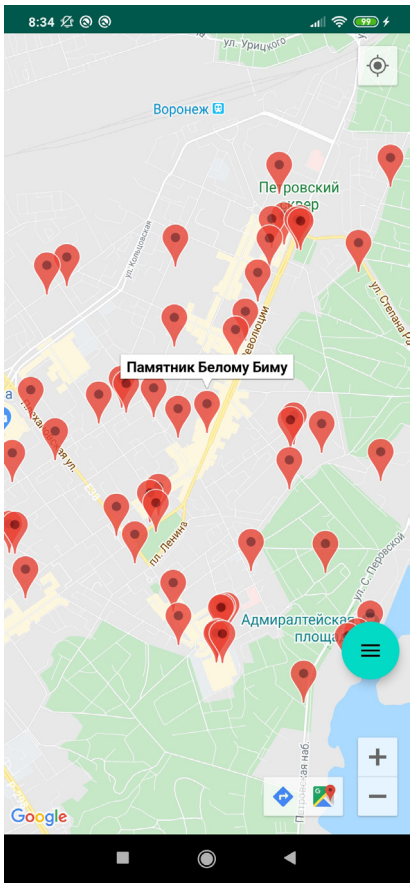


Рис. 5. Карта с достопримечательностями

Для демонстрации были выбраны следующие достопримечательности:

1. Памятник Андрею Платонову;
2. Памятник А. С. Пушкину;
3. Памятник Петру I;
4. Музей Истории ВГУ;
5. Советская площадь.

В качестве пункта отправления выбран Памятник Андрею Платонову. Выбор достопримечательностей показан на рис. 6.

В результате работы программы с использованием жадного алгоритма, получены следующие результаты, представленные на рис. 7.

Оптимальный маршрут через выбранные достопримечательности выглядит так:

1. Памятник Андрею Платонову;
2. Памятник Петру I;

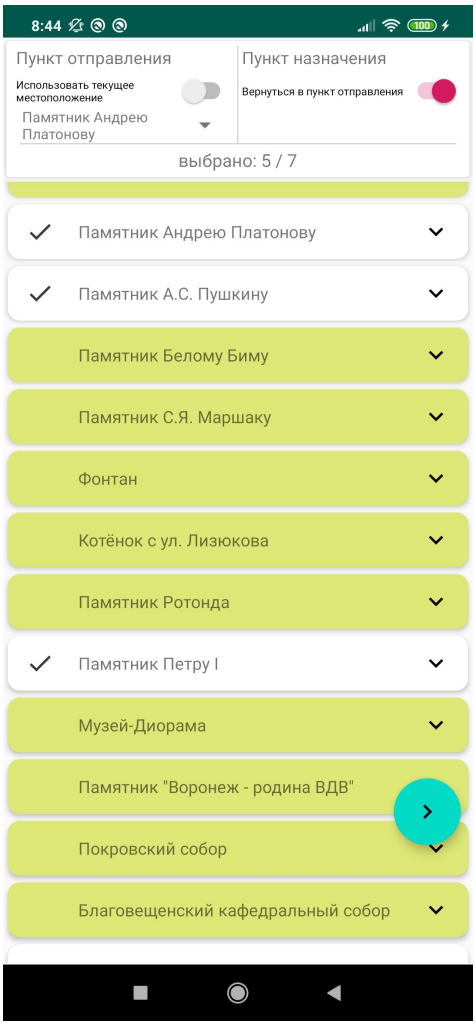


Рис. 6. Выбор достопримечательностей

Рис. 7. Результаты работы программы, полученные «жадным» алгоритмом

3. Советская площадь;
4. Памятник А.С. Пушкину;
5. Музей Истории ВГУ;
6. Памятник Андрею Платонову.

Суммарная длина маршрута составляет 6681 метр, а время, за которое его можно преодолеть – 23 минуты.

При использовании муравьиного алгоритма (рис. 8) приложение предоставляет на выбор три наилучших маршрута.

Первый маршрут является самым коротким:

1. Памятник Андрею Платонову;
2. Памятник Петру I;
3. Советская площадь;
4. Музей Истории ВГУ;
5. Памятник А.С. Пушкину
6. Памятник Андрею Платонову.

Его длина маршрута составляет 6630 метров, и можно пройти – 22 минуты. Длина второго маршрута – 6681 метр. Третий маршрут – 6807 метров.

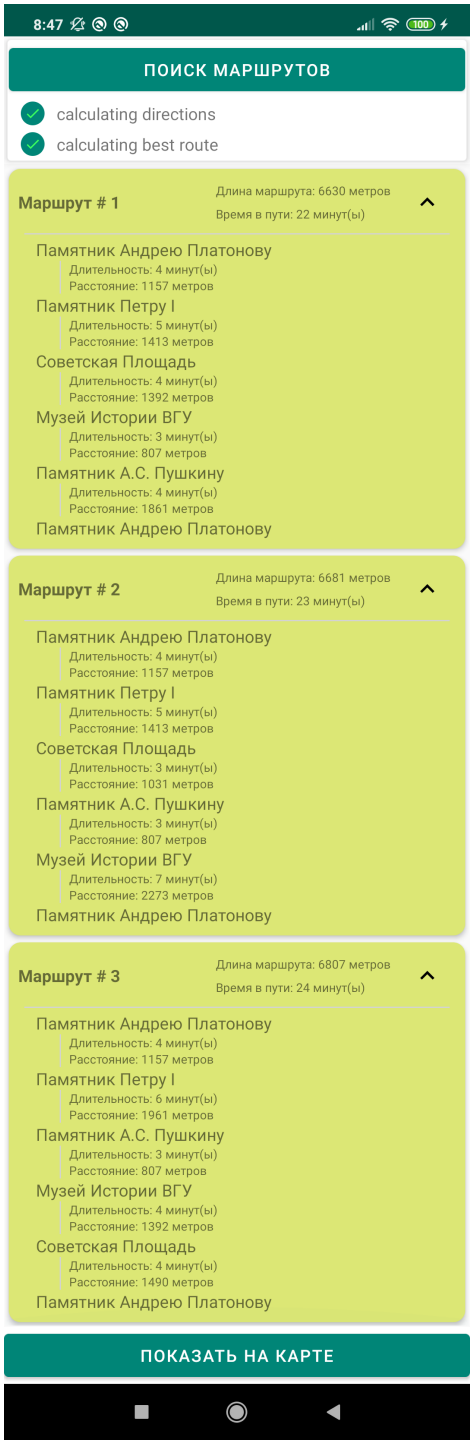


Рис. 8. Результаты работы программы, полученные муравьиным алгоритмом

Как видно из результатов, муравьиный алгоритм построил кратчайший маршрут длиной в 6630 метров, а маршрут с результатом 6681 метр, полученный жадным алгоритмом, не является оптимальным.

Литература

1. *Штовба, С. Д.* Муравьиные алгоритмы / С. Д. Штовба // Exponenta Pro. Математика в приложениях. – 2003. – №4. – С. 70–75.

2. *Кочегурова, Е. А.* Алгоритм муравьиных колоний для задачи проектирования рациональных маршрутных сетей городского пассажирского транспорта / Е. А. Кочегурова [и др.] // Вестник СибГУТИ. – 2014. – № 3. – С. 89–100.

3. *Курейчик, В. М.* О некоторых модификациях муравьиного алгоритма / В. М. Курейчик, А. А. Кажаров // Известия Южного федерального университета. Технические науки. – 2008. – № 4. – С. 7–12.

4. Эволюционные вычисления: электронный курс. – Режим доступа: https://www.intuit.ru/studies/courses/14227/1284/info.

5. Муравьиный алгоритм. – Режим доступа: https://ru.wikipedia.org/wiki/Муравьиный_алгоритм.

6. Задача коммивояжёра. – Режим доступа: https://ru.wikipedia.org/wiki/Задача_коммивояжёра.

7. Android разработка – с нуля до профессионала. Полный курс. – Режим доступа: https://www.udemy.com/course/android-kak-po-notam-a.

8. Общие сведения о платформе Android. – Режим доступа: https://developer.android.com/guide.

9. Google Maps & Google Places Android Course. – Режим доступа: https://www.youtube.com/watch?v=OknMZUnTyds&list=PLgCYzUzKIBE-vInwQhGSdnbyJ62nixHCt.

10. Google Maps Platform Documentation. – Режим доступа: https://developers.google.com/maps/documentation.

Смирнов Александр Сергеевич – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: smirnov@amm.vsu.ru

УДК 004.853

DROPOUT КАК МЕТОД РЕШЕНИЯ ПРОБЛЕМЫ ПЕРЕОБУЧЕНИЯ В ГЛУБОКИХ НЕЙРОННЫХ СЕТЯХ

Н. П. Строков, С. Ю. Болотова

Воронежский государственный университет

Введение

Переобучение является одной из важнейших проблем глубоких нейронных сетей (Deep Neural Networks, DNN). Оно выражается в том, что хорошо объясняет только примеры из обучающей выборки, адаптируясь к обучающим примерам, вместо того чтобы учиться классифицировать примеры, не участвовавшие в обучении, теряя способность к обобщению. За последние годы было предложено множество решений этой проблемы, но Dropout превзошел все остальные, благодаря своей простоте и прекрасным практическим результатам.

1. Применение Dropout

Когда большая нейронная сеть с прямой связью обучается на небольшом обучающем наборе, она плохо справляется с данными выдержанных испытаний. Переоснащение сокращается за счет случайного исключения половины детекторов функций в каждом учебном случае. Это предотвращает сложные коадаптации, в которых детектор признаков полезен только в контексте нескольких других детекторов специфических особенностей. Вместо этого каждый нейрон учится обнаруживать особенность, которая обычно полезна для получения правильного ответа, учитывая комбинаторно большое разнообразие внутренних контекстов, в которых он должен работать. Случайное выпадение дает значительные улучшения для многих контрольных задач. Искусственная нейронная сеть с прямой связью использует слои нелинейных «скрытых» единиц между входами и выходами. Адаптируя весовые коэффициенты на входящих соединениях этих скрытых блоков, он изучает детекторы функций, которые позволяют ему прогнозировать правильный выходной сигнал при заданном входном векторе.

На рис.1 слева представлена нейронная сеть после Dropout, справа – та же сеть до Dropout.

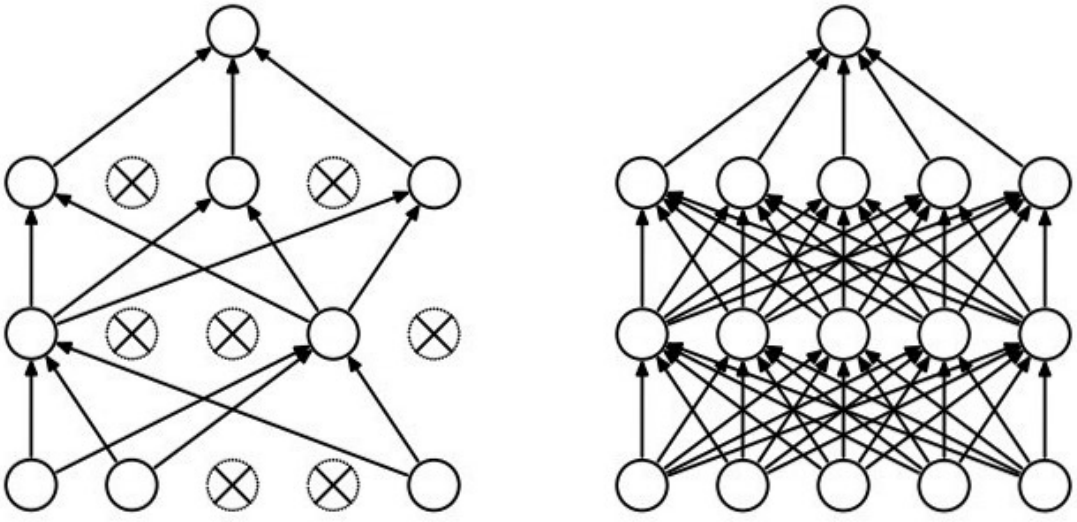


Рис. 1

Главная идея Dropout в том, чтобы вместо обучения одной DNN, обучить ансамбль нескольких DNN, а затем усреднить полученные результаты. Исключение нейрона означает, что при любых входных данных или параметрах он возвращает 0. Такие нейроны не вносят свой вклад в процесс обучения ни на одном из этапов алгоритма обратного распространения ошибки (backpropagation); поэтому исключение хотя бы одного из нейронов равносильно обучению новой нейронной сети.

Dropout выключает нейроны с вероятностью p и оставляет их включенными с вероятностью $q = 1 - p$. Вероятность выключения каждого нейрона одинакова. При условии, что $h(x) = xW + b$ – линейная проекция входного d_i-мерного вектора x на d_h-мерное пространство выходных значений; a_h – функция активации, применение Dropout к данной проекции на этапе обучения можно представить как измененную функцию активации:

$$f(h) = Da(h),$$

где $D = (X_1, ..., X_{d_h})$ – d_h-мерный вектор случайной величины X_i, распределенных по закону Бернулли. X_i имеет следующее распределение вероятностей:

$$f(k; p) = \begin{cases} p, & if \quad k = 1 \\ 1 - p, & if \quad k = 0 \end{cases},$$

где k – все возможные выходные значения.

Эта случайная величина идеально соответствует Dropout, примененному к одному нейрону. Так как на этапе обучения нейрон остается в сети с вероятностью q, на этапе тестирования нам необходимо эмулировать поведение ансамбля нейронных сетей, использованного при обучении. Для этого предлагается на этапе тестирования умножить функцию активации на коэффициент q. Таким образом:

На этапе обучения: $O_i = X_i a(\sum_{k=1}^{d_i} w_k x_k + b)$.

На этапе тестирования: $O_i = qa(\sum_{k=1}^{d_i} w_k x_k + b)$.

Dropout множества нейронов

Слой h из n нейронов на отдельном шаге этапа обучения можно рассматривать как ансамбль из n экспериментов Бернулли с вероятностью успеха p. На выходе слоя h мы получаем следующее количество исключенных нейронов:

$$Y = \sum_{i=1}^{d_h} (1 - x_i).$$

Так как каждый нейрон представлен в виде случайной величины, распределенной по закону Бернулли, и все эти величины независимы, общее число исключенных нейронов будет такой же случайной величиной, но имеющей биномиальное распределение:

$$YBi(d_h, p),$$

где вероятность k успешных событий за n попыток характеризуется плотностью распределения:

$$f(k, n, p) = \binom{n}{k} p^k (1 - p)^{n-k}$$

$p^k (1 - p)^{n-k}$ – вероятность получения последовательности k успешных событий за n попыток, и $n - k$ неуспешных. $\binom{n}{k}$ – биномиальный коэффициент, используемый для расчета количества возможных успешных последовательностей.

Заключение

Глубокие нейронные сети с большим числом параметров являются очень мощными инструментами машинного обучения. Переобучаемость в них является серьезной проблемой. Крупные сети достаточно медлительны, требуя использования нескольких различных больших нейронных сетей во время тестирования. Даже если имеется возможность обучения нескольких различных больших сетей, использование их все время проведения испытания является недопустимым в приложениях, где важно время реакции. Dropout является методом, который решает обе эти проблемы. Данный метод предотвращает переобучение и обеспечивает способ объединения приблизительно экспоненциального количества различных сетевых архитектур нейронных сетей.

Список литературы

1. *Hinton, G. E.* Improving neural networks by preventingco-adaptation of feature detectors / G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov // Neural and Evolutionary Computing. – 2012. – С. 1–18

2. *Rumelhart, D. E.* Nature VOL. 323 / D. E. Rumelhart, G. E. Hinton, R. J. Williams // Letters to nature. – 1986. – С. 533–536.

3. *Ciresan, L. M. G. D. C.* Neural Computation / L. M. G. D. C. Ciresan, U. Meier, J. Schmidhuber // Neural Computation. – 2010. – 3223 с.

Строков Никита Павлович – студент 4-го курса кафедры Математического Обеспечения ЭВМ Воронежского государственного университета. E-mail: flasher80@yandex.ru

Болотова Светлана Юрьевна (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: Bolotova.svetlana@gmail.com

УДК 004.92

ПОСТРОЕНИЕ ТЕТРАЭДРИЧЕСКОЙ СЕТКИ НА ОСНОВЕ ТРИАНГУЛИРОВАННОЙ МОДЕЛИ

А. С. Сырых

Воронежский государственный университет

Введение

Тетраэдрические сетки используются в ряде областей, включая деформацию объектов, твердотельное моделирование и виртуальную хирургию. Задача построения правильной и качественной сетки, описывающей произвольную область, занимает много времени, поэтому всё больше и больше появляется интерес к автоматическому созданию сеток. Одной из сложностей построение сетки является потребность корректно описать форму объекта. Однако у него могут быть острые углы и складки и в виртуальной хирургии, например, нельзя жертвовать качеством тетраэдров.

Постановка задачи

Дана трёхмерная триангулированная модель (рис. 1). Целью данной статьи является построения тетраэдрической сетки из данной модели с хорошими двугранными углами в диапазоне от 10,7° и 164,8°. Таким образом на выходе необходимо получить тетраэдрическую сетку, правильно описывающую форму модели, внутри и на границе которой находятся тетраэдры пригодные для симуляции. На рис. 2 показан пример возможной тетраэдрической сетки.

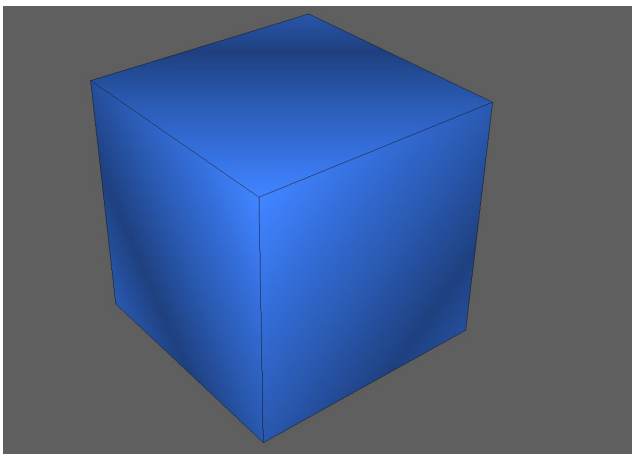


Рис. 1. Исходная модель

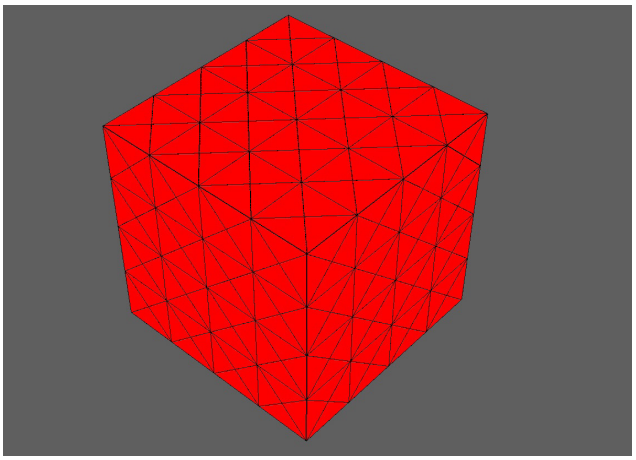


Рис. 2. Возможная тетраэдрическая сетка

1. Алгоритм построения тетраэдрической сетки для трёхмерной модели

Вначале вычисляется функция $f : R^3 \to R,$ которая имплементирует принадлежность точек тетраэдра модели. Если значение функции f в точке отрицательно, то считается, что точка лежит за пределами триангулированной модели, если положительна – внутри, а если значение равно нулю, то точка лежит на границе модели. Данную функцию можно назвать упрощённой функцией расстояния со знаком [3, 4]. Существует множество методов построения тетраэдри-

ческих сеток: разбиение Делоне [1], метод на основе Octree дереве [2, 5], метод решёток [1, 5]. В данной статье используется метод решёток, поскольку у него есть несколько преимуществ перед остальными методами. Например, преимущество перед методом на основе Octree заключается в следующем: нет смысла объединять решётки до однородного уровня. Все тетраэдры имеют одинаковую структуру, что позволяет управлять качеством тетраэдров путём увеличения или уменьшения количества решёток в основном кубе, ограничивающем модель. Преимущество перед разбиением Делоне: при методе решёток все тетраэдры получаются одинаковые с двугранными углами в диапазоне между 40–90°. В разбиение Делоне возникает проблема вставки новой вершины: если новая вершина будет слишком близко к другой, то получиться слишком маленький двугранный угол, который может поломать всю симуляцию.

1 шаг. Построение bcc решётки для модели

Необходимо построить ограничивающий куб модели (bounding box) после этого разбить этот куб на подкубы. Далее необходимо заполнить все подкубы тетраэдрами (рис. 3). В одном кубе содержится 24 тетраэдра. Всё семейство этих подкубов называется BCC lattice (body-centered cubic lattice) (рис. 4) .

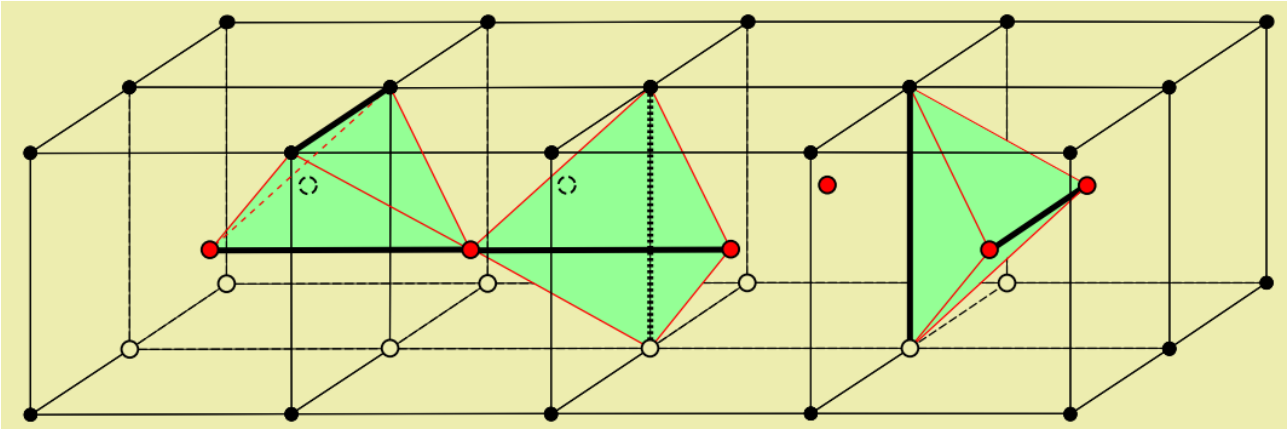


Рис. 3. Построение BCC lattice

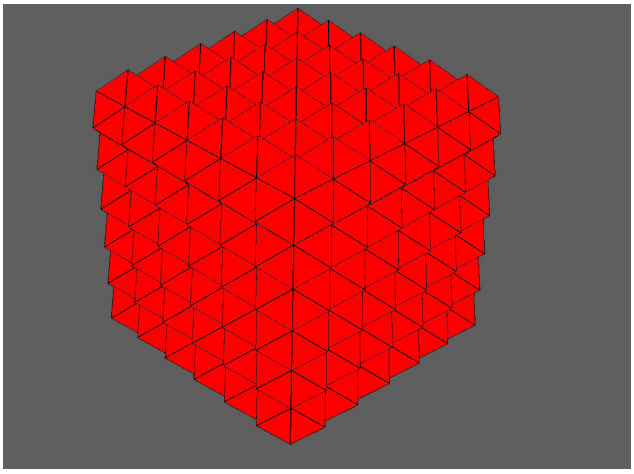


Рис. 4. (1 шаг) Построение BCC-lattice

2 шаг. Удаление внешних тетраэдров

Для удаления внешних тетраэдров необходимо построить signed distance field (функцию расстояния со знаком). Существует множество способов построения sdf и определения положения точки относительно полигональной модели. Одним из способов определения является метод на основе поиска ближайшего треугольника. Он состоит из следующих шагов:

1. Найти ближайший треугольник.
2. Вычислить его нормаль и вектор разности между просматриваемой точкой и точкой на модели.
3. Посчитать скалярное произведение вектора и нормали.
4. Взять получившееся число с противоположным знаком.
5. Если оно положительно, то точка находиться внутри модели, если отрицательно – снаружи и равно нулю – на границе модели.

После просмотра всех точек и маркирования можно удалить те тетраэдры, точки которых помечены как внешние (рис. 5).

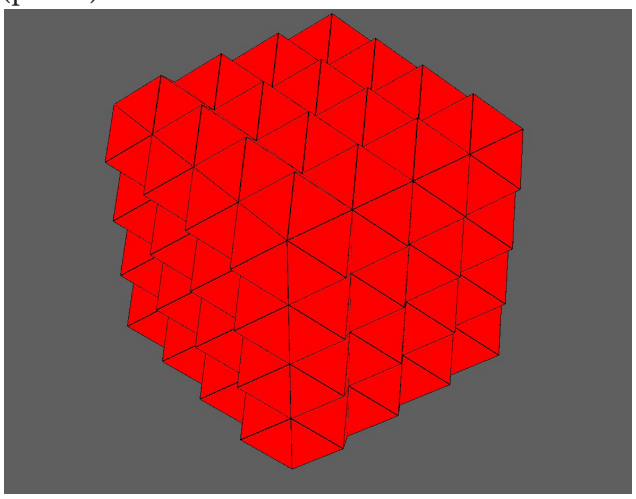


Рис. 5. (2 шаг) Отбрасывание внешних тетраэдров

3 шаг. Поиск точек среза

Необходимо задать значение точки среза α в диапазоне от [0,0;0,5]. Данный шаг использует следующее правило: если точка среза лежит на краю сетки и это расстояние меньше чем

$$\alpha \times \text{длина ребра} \times \text{двугранный угол при этом ребре} \quad (1)$$

то можно считать, что данная точка нарушает это правило и она должна быть смещена до уровня границы модели (рис. 6). После первой итерации данного шага может получиться так, что все точки не были помещены на модель (т. е. нарушают данное правило). Вторая итерация данного алгоритма поможет решить такую проблему. Но это порождает следующие проблемы: могут быть затронуты уже правильно расположенные точки и процесс может повторяться до бесконечности. Поэтому необходимо выбрать более слабую границу с двугранными углами.

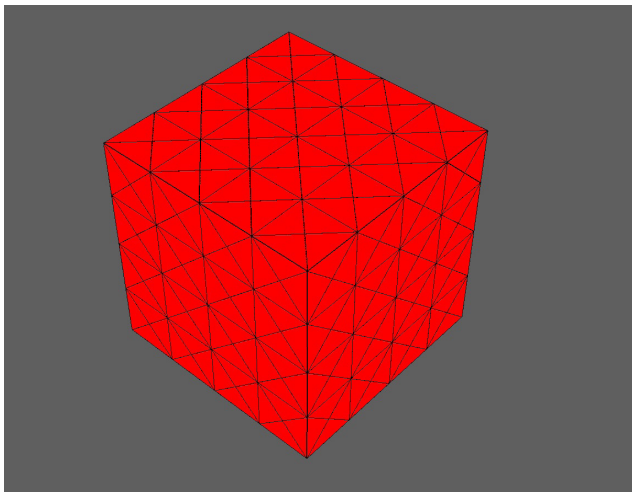


Рис. 6. Поиск точек среза и вдавливание тетраэдров

Пример работы ПО

На рис. 7 показана тетраэдрическая сетка зайца
На рис. 8 показана тетраэдрическая сетка сферы
На рис. 9 показана тетраэдрическая сетка торуса (вид на срезе)

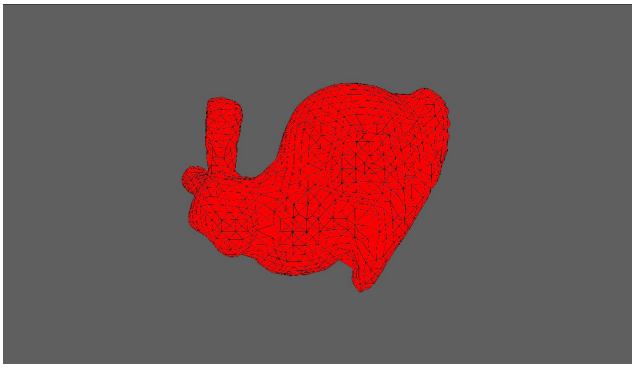


Рис. 7. Тетраэдрическая сетка модели «Bunny»

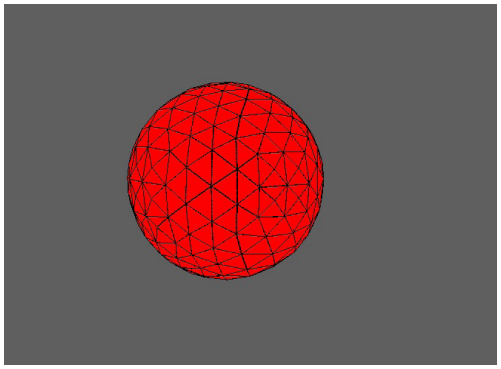


Рис. 8. Тетраэдрическая сетка модели «Сфера»

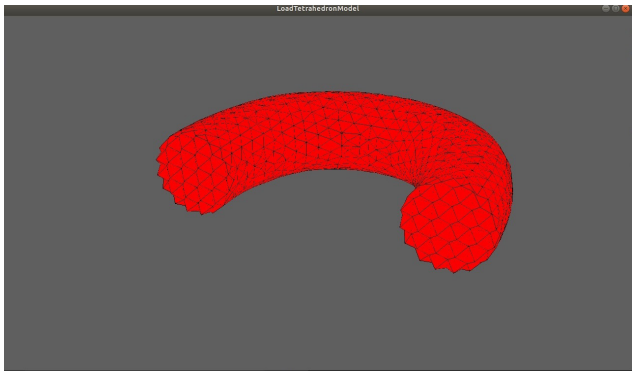


Рис. 9. Тетраэдрическая сетка модели «Торус» (вид на срезе)

Вывод

Таким образом по полученным данным можно считать, что была получена тетраэдрическая сетка на основе триангулированной модели с корректным описанием границ модели и

двугранными углами в диапазоне от 10,7° и 164,8°. Дальнейшей целью исследования является поиск и модификация более оптимального разбиения, требующего меньшего количества вычислительных мощностей и более гибким в динамическом перестроении, чем предоставленный в данной статье.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. *Shewchuk J.* Constrained Delaunay Tetrahedralizations and Provably Good Boundary Recovery. – 11th Int. Meshing Roundtable. – 2002.

2. *Radovitzky R. A., Ortiz M.* Tetrahedral Mesh Generation Based in Node Insertion in Crystal Lattice Arrangements and Advancing-Front Delaunay Triangulation // Comput. Meth. in Appl. Mech. and Eng. – 2000. – V. 187. – P. 543–569.

3. *Sadarjoen I. A., Post F. H.* Deformable Surface Techniques for Field Visualization // Eurographics. – 1997. – P. 109– 116.

4. *Klaus E., Markus H., Joe K., Christof R.-S., Daniel W.* Real-Time Volume Graphics A K Peters. – P. 1–515.

5. *Crawford Doran.* Acute Lattices and Feature Matching // B. Software Engineering. – University of Waterloo, 2011.

Сырых Александр Сергеевич – студент 3-его курса кафедры МО ЭВМ Воронежского государственного университета. E-mail: zsyryhz@mail.ru

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 517.9

К РАЗРЕШИМОСТИ УРАВНЕНИЙ В БАНАХОВОМ ПРОСТРАНСТВЕ

А. С. Сырых, Д. Г. Усков

Воронежский государственный университет

В методе подобных операторов [1–4] важными являются вопросы разрешимости соответствующих операторных уравнений. Рассмотрим далее уравнение того же типа, что и возникающее при применении метода К. О. Фридрихса подобных операторов (см. [1]). Пусть $\mathscr{X}$ – банахово пространство и $\mathrm{BL}(\mathscr{X})$ – банахова алгебра всех ограниченных линейных операторов, действующих в $\mathscr{X}$ с нормой $\|X\| = \sup\limits_{\|x\|\le 1}\|Xx\|$, $x\in\mathscr{X}$, $X\in BL(\mathscr{X})$. В банаховом пространстве $\mathscr{X}$ рассматривается уравнение

$$x = Ax + b, \tag{1}$$

где $A\in BL(\mathscr{X})$, $x\in\mathscr{X}$, $b\in\mathscr{X}$. Уравнение метода Фридрихса подобных операторов [1] относится к уравнению рассматриваемого типа.

Определение 1 [5]. Для любого $b\in\mathscr{X}$ величина $r(b, A) = \overline{\lim\limits_{n\to\infty}}\sqrt[n]{\|A^n b\|}$ называется спектральным радиусом вектора b относительно оператора A.

Напомним (см. [6]), что для оператора $B\in BL(\mathscr{X})$ его спектральным радиусом $r(B)$ называется $\overline{\lim\limits_{n\to\infty}}\|B^n\|^{\frac{1}{n}} = r(B)$. При этом имеет место оценка $r(B)\le\|B\|$.

Лемма 1. $r(A) = \sup\limits_{\|b\|\le 1} r(b, A)$, *где* $A\in BL(\mathscr{X})$, $b\in\mathscr{X}$.

Доказательство. Действительно,

$$r(A) = \overline{\lim_{n\to\infty}}\|A^n\|^{\frac{1}{n}} = \lim_{n\to\infty}\left\|\sup_{\|b\|\le 1} A^n b\right\|^{\frac{1}{n}} = \sup_{\|b\|\le 1} r(b, A).$$

Следствие. *Для* $A\in BL(\mathscr{X})$ *и* $b\in\mathscr{X}$ $r(b, A)\le r(A)$.

Лемма 2. *Для любых двух элементов* $a, b\in\mathscr{X}$ *и* $A\in BL(\mathscr{X})$ *имеет место оценка*

$$r(a+b, A)\le r(a, A) + r(b, A).$$

Доказательство. Имеем

$$r(a+b, A) = \overline{\lim_{n\to\infty}}\|A^n(a+b)\|^{\frac{1}{n}} = \overline{\lim_{n\to\infty}}\|A^n a + A^n b\|^{\frac{1}{n}} \le \overline{\lim_{n\to\infty}}\|A^n a\|^{\frac{1}{n}} + \overline{\lim_{n\to\infty}}\|A^n b\|^{\frac{1}{n}} = r(a, A) + r(b, A).$$

Лемма 3. *Пусть* $A\in BL(\mathscr{X})$ *и* $U\in BL(\mathscr{X})$ *перестановочен с* A, *то есть* $AU = UA$. *Тогда* $r(Ub, A)\le r(b, A)$.

Доказательство. Действительно,

$$r(Ub, A) = \overline{\lim_{n\to\infty}}\left(A^n Ub\right)^{\frac{1}{n}} \le \overline{\lim_{n\to\infty}}\|U\|^{\frac{1}{n}}\cdot\left(A^n b\right)^{\frac{1}{n}} \le r(b, A).$$

Теорема 1. *Пусть*

$$r(b, A) < 1. \tag{2}$$

Тогда уравнение (1) имеет решение, задаваемое формулой

$$x = \sum_{k=0}^{\infty} A^k b = b + Ab + A^2 b + \ldots + A^n b + \ldots, \tag{3}$$

причём выписанный ряд является абсолютно сходящимся.

Результат теоремы 1 получается непосредственной подстановкой ряда (3) в уравнение (1), а условие (2) обеспечивает абсолютную сходимость выписанного ряда.

Замечание 1. Теорема 1 гарантирует единственность решения уравнения (1) (единственность неподвижной точки отображения $f:\mathscr{X}\to\mathscr{X}$, где $f(x)=Ax+b$). Отметим, что аффинное отображение f является сжимающим тогда и только тогда, когда $\|A\|<1$.

Отметим, что если выполнено условие (2) для любого вектора $b\in\mathscr{X}$, тогда оператор $I-A$ обратим в $\mathscr{X}$ и обратный к нему оператор $(I-A)^{-1}$ представим в виде ряда

$$(I-A)^{-1}=\sum_{k=0}^{\infty}A^k.$$

Теорема 2. *Пусть выполнено одно из следующих двух условий:*

1. $\lim\limits_{n\to\infty}\dfrac{\|A^{n+1}y\|}{\|A^n y\|}=0$, *для любого* $y\in\mathscr{X}$;
2. $\overline{\lim\limits_{n\to\infty}}\|A^n y\|^{\frac{1}{n}}=0$, *для любого* $y\in\mathscr{X}$.

Тогда уравнение (1) имеет решение, задаваемое формулой (3).

Также отметим, что в стандартном методе подобных операторов, кроме линейного уравнения Фридрихса, обычно используются различные нелинейные уравнения (см. [7–10]).

Литература

1. *Friedrichs, K. O.* Lectures on advanced ordinary differential equations / K. O. Frie-drichs. – New York : Notes by P. Berg, W. Hirsch, P. Treuenfels, Gordon and Breach Science Publishers, 1965. – 205 p.
2. *Баскаков, А. Г.* Гармонический анализ линейных операторов / А. Г. Баскаков. – Воронеж: Изд-во ВГУ, 1985. – 165 с.
3. *Баскаков, А. Г.* Методы абстрактного гармонического анализа в теории возму-щений линейных операторов / А. Г. Баскаков // Сиб. матем. журн. – 1983. – Т. 24, № 1. – С. 27–39.
4. *Баскаков, А. Г.* Замена Крылова-Добролюбова в теории возмущений линейных операторов / А. Г. Баскаков // Укр. матем. журн. – 1984. – Т. 36, № 5. – С. 606–611.
5. *Данфорд, Н.* Линейные операторы: в 3-х т. / Н. Данфорд, Д. Т. Шварц. – М. : Мир, 1974. – Т. 3: Спектральные операторы. – 661 с.
6. *Садовничий, В. А.* Теория операторов / В. А. Садовничий. – М. : Высшая шко-ла, 1999. – 368 с.
7. *Баскаков, А. Г.* Метод подобных операторов в спектральном анализе оператора Хилла с негладким потенциалом. / А. Г. Баскаков, А. В. Дербушев, А. О. Щербаков // Изв. РАН. Сер. Матем. – 2011. – Т. 75, № 3. – С. 3–28.
8. *Баскаков, А. Г.* Метод подобных операторов в спектральном анализе оператора Хилла с негладким потенциалом / А. Г. Баскаков, Д. М. Поляков // Матем. сб. – 2017. – Т. 208, № 1. – С. 3–47.
9. *Baskakov, A. G.* Similarity techniques in the spectral analysis of perturbed operator matrices / A. G. Baskakov, I. A. Krishtal, N. B. Uskova // J. Math. Anal. and Appl. – 2019. – V. 477. – P. 930–960.
10. *Баскаков, А. Г.* Метод Фурье для дифференциальных уравнений первого порядка к группам операторов / А. Г. Баскаков, Н. Б. Ускова // Уфимский матем. журн. – 2018. – Т. 10, № 3. – С. 11–13.

Сырых Александр Сергеевич – студент 3 курса кафедры математического обеспечения ЭВМ факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: zsyryhz@mail.ru

Усков Даниил Геннадьевич – студент 3 курса кафедры математического обеспечения ЭВМ факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: uskov.dan@mail.ru

Баскаков Анатолий Григорьевич (научный руководитель) – д-р. физ.-мат. наук, проф., профессор кафедры системного анализа и управления факультета прикладной математики, информатики и механики Воронежского государственного университета. E-mail: anatbaskakov@yandex.ru

УДК 004.75

АНАЛИЗ КАЧЕСТВА ВОЖДЕНИЯ ТС С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ ОБЛАЧНЫХ ТЕХНОЛОГИЙ

В. О. Тарасова, Е. В. Трофименко

Воронежский государственный университет

Введение

В настоящее время облачные технологии являются одним из наиболее динамично развивающихся направлений в сфере информационных технологий.

Облачные технологии не только обеспечивают сетевой доступ к вычислительным ресурсам – серверам, устройствам хранения данных, приложениям и сервисам – но также эффективны для обработки и анализа больших данных. Таким образом, данные для анализа качества вождения транспортного средства, получаемые с сенсоров мобильного устройства и подлежащие обработке в реальном времени, успешно анализируются с помощью облачных технологий.

Наиболее результативна для указанных целей технология Apache Spark, поскольку эта технология позволяет разработать эффективную систему анализа, преобразования и дальнейшей классификации входящих данных с датчиков мобильного устройства о передвижении автомобиля по категориям качества вождения.

1. Постановка задачи

Основная задача разработки – предоставление пользователю информации о качестве его вождения (в том числе, нарушениях) в реальном времени с помощью анализа данных, получаемых с мобильного устройства пользователя.

Для получения входных данных предполагается использовать анализ алгоритмов снижения погрешности исходных данных с датчиков смартфона (акселерометр, гироскоп, GPS и магнитометр). На основе выбора наиболее подходящего алгоритма предполагается реализовать предварительную обработку, преобразование и группировку данных для дальнейшего анализа.

Таким образом, результатом анализа данных будет выступать предоставление пользователю сведений о его поездках, совершенных маневрах, таких как ускорение, торможение, поворот, превышение скорости, а также степень резкости и опасности данных маневров. Степень резкости и опасности совершенных пользователем (водителем) действий определяется на основе анализа и установки порога чувствительности для каждого из таких действий.

2. Решение задачи

2.1. Алгоритм сбора данных на смартфоне

ОС Android представляет данные с датчиков в виде набора цифровых значений, используя стандартную трех осевую систему координат (рис. 1).

Для решения текущей задачи нам понадобятся следующие датчики (табл. 1) и система GPS:

1. Акселерометр – измеряет ускорение, которое прилагается к устройству.

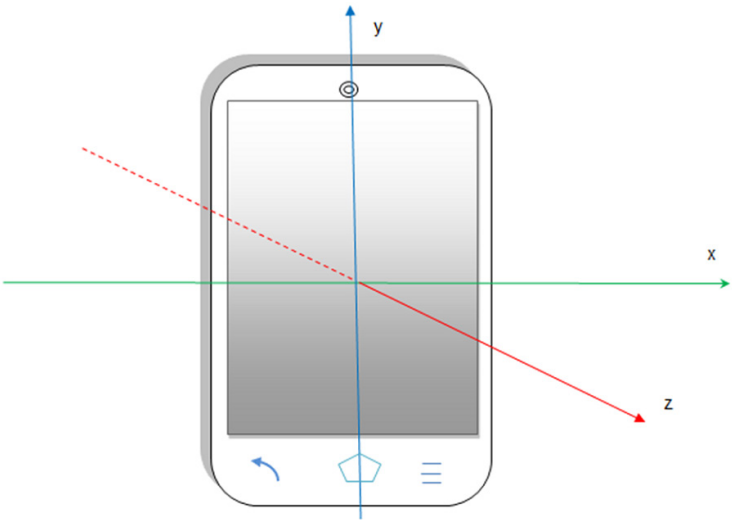


Рис. 1. Оси координат устройства

2. Гироскоп – измеряет скорость вращения устройства вокруг осей x, y, z. Вращение вокруг оси против часовой стрелки будет давать положительные значения данных гироскопа, а по часовой стрелке – отрицательные.

3. Магнитомерт – определяет текущие показатели магнитного поля в микротеслах по трем осям, то есть выполняет функцию компаса.

4. GPS – является системой, основанной на получении спутниковых данных, которая выдаёт информацию о географическом положении по всему миру. Для решения задачи потребуются данные долготы, широты, скорости.

Таблица 1

Значения, возвращаемые датчиками

Тип датчика	Значение	Описание
TYPE_ACCELEROMETER	value[0]: ось X value[1]: ось Y value[2]: ось Z	Ускорение (м/с2) по трём осям
TYPE_LINEAR_ ACCELERATION	value[0]: ось X value[1]: ось Y value[2]: ось Z	Линейное ускорение (м/с2) по трём осям без учёта силы тяжести
TYPE_GYROSCOPE	value[0]: ось X value[1]: ось Y value[2]: ось Z	Скорость вращения (рад/с) по трём осям
TYPE_MAGNETIC_FIELD	value[0]: ось X value[1]: ось Y value[2]: ось Z	Внешнее магнитное поле (мкТл)

На стороне смартфона собираются данные для последующего анализа трех типов: ускорение автомобиля в момент времени, угол поворота автомобиля, координаты и скорость автомобиля. Каждый тип данных обрабатывается в отдельных потоках приложения. На протяжении одной поездки в течение секунды времени накапливаются показатели ускорения, угла поворота, координат и скорости автомобиля. Вычисляется среднее значение по этим данным в секунду времени. Далее для объединения этих групп данных в дальнейшем для анализа на сервере ключом объединения будет являться показатель времени и идентификатор поездки.

Для определения ускорения, торможения автомобиля в момент времени подойдет датчик TYPE_LINEAR_ACCELERATION – это виртуальный датчик, использующий показания акселерометра. Для определения ориентации в пространстве в ОС Android есть стандартные средства, с помощью которых возможно получить данные о трех углах наклона. Решение стандартными средствами дает сильную погрешность в полученных данных, которые будут давать неверные результаты при их анализе.

Существует множество алгоритмов для снижения погрешности и фильтрации данных. В результате анализа существующих алгоритмов [1–3] были выделены следующие: фильтр Калмана, фильтр Маджвика, комплементарный фильтр.

Результат анализа работы этих алгоритмов показал, что все три фильтра взаимодействуют с данными акселерометра, гироскопа и магнитометра, объединяя их в вычислениях для получения более точных углов наклона (крен, тангаж и азимут). Для решения поставленной задачи наиболее подходящим является комплементарный фильтр, так как применение данного фильтра не требует от устройства большой вычислительной мощности.

2.2. Комплементарный фильтр

Угол наклона относительно осей можно вычислять с помощью гироскопа, либо с помощью акселерометра. Но в случае гироскопа точность таких расчетов снижается из-за дрейфа нуля и ошибок интегрирования по времени. В случае же акселерометра слишком велика чувствительность к внешним воздействиям. Комплементарный фильтр позволяет объединить показания этих двух датчиков для устранения их недостатков. Чтобы избежать как дрейфа гироскопа, так и ориентации с помехами, выход гироскопа применяется только для изменения ориентации в короткие промежутки времени, в то время как данные магнитометра / акселерометра используются в качестве вспомогательной информации в течение длительных периодов времени. Это эквивалентно низкочастотной фильтрации сигналов акселерометра и датчика магнитного поля и высокочастотной фильтрации сигналов гироскопа. На рис. 2 приведен общий датчик слияния и фильтрации.

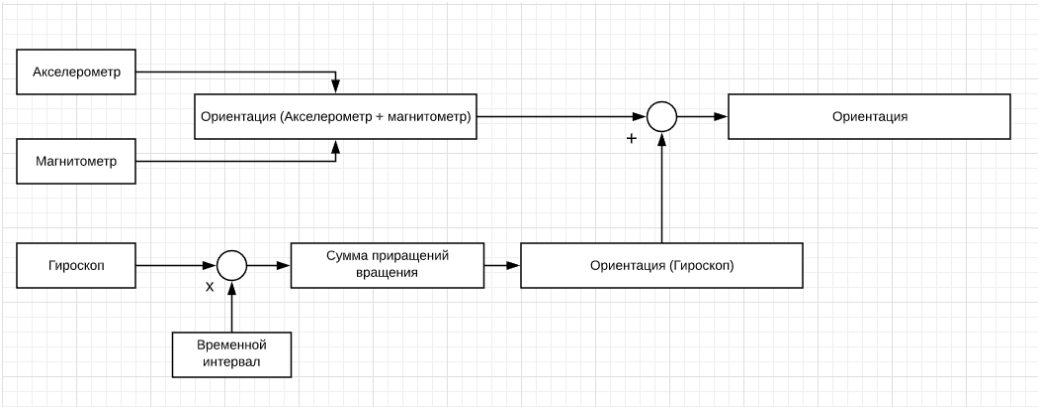


Рис. 2. Схема вычисления ориентации

Искомый угол наклона будет вычисляться по формуле:

$$a(t) = (1 - K) \cdot (a(t-1) + gx \cdot dt) + K \cdot acc,$$

где $a(t)$ – искомый угол наклона, учитывающий показания акселерометра; $a(t-1)$ – угол тела в предыдущий момент времени; gx – скорость вращения тела вокруг оси X; dt – время, которое прошло с момента предыдущего вычисления угла a; acc – значение угла наклона, полученное при помощи акселерометра; K – коэффициент комплементарного фильтра. Итоговая величина угла наклона представляет собой сумму интегрированного значения гироскопа и мгновен-

ного значения акселерометра. Главная задача комплементарного фильтра здесь в том, чтобы с помощью показаний акселерометра нивелировать дрейф нуля гироскопа и ошибки дискретного интегрирования. На рис. 3 приведен график результата работы комплементарного фильтра, где синий график – это угол, вычисленный по показаниям гироскопа, красный – угол по акселерометру, зеленый – угол, вычисленный при помощи комплементарного фильтра.

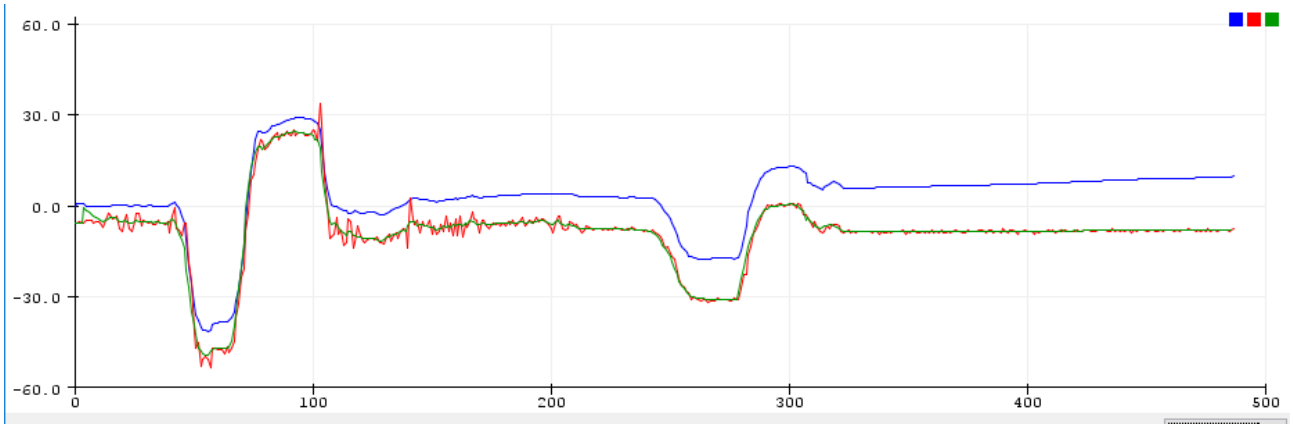


Рис. 3. Результат работы комплементарного фильтра

2.3. Анализ данных на сервере

Каждая запись, характеризующая движение автомобиля, ускорение, поворот, координаты, скорость в секунду времени отправляется на сервер для дальнейшей обработки и анализа. Данные водителя будут поступать в промежуточный кластер, работающий под управлением Apache Kafka. Apache Kafka – это распределенный программный брокер сообщений, который сохраняет поступающие сообщения на определенный промежуток времени для последующей их обработки. В качестве системы для дальнейшей потоковой обработки данных будет использована Apache Spark с ее расширением Spark Streaming, которое позволяет осуществлять отказоустойчивую обработку потока данных с высокой пропускной способностью.

Для решения поставленной задачи были разработаны следующие модули по обработке и анализу данных:

1. Модуль анализа данных с датчика акселерометр для определения ускорения автомобиля и его степени опасности.

2. Модуль анализа данных с датчика акселерометр для определения торможения автомобиля и его степени опасности.

3. Модуль анализа данных, полученных в результате применения комплементарного фильтра, то есть данных о повороте автомобиля и его степени опасности.

4. Модуль анализа данных, полученных системой GPS, для определения превышения скорости.

Пороговые значения чувствительности для каждого из таких маневров установлены на основании собранных эмпирических данных. В табл. 2, табл. 3, табл. 4 приведены пороговые значения чувствительности ускорения, торможения и поворота соответственно для легкового автомобиля.

Для определения превышения скорости по геопозиции и определения местоположения нарушения использовалось стороннее API.

Для построения алгоритма решения такой задачи классификации был выбран метод дерево решений – один из метод автоматического анализа данных. В результате анализа полученные данные о нарушениях сохраняются в базу данных Postgresql, которые будут отображены

водителю на экране смартфона в виде: адрес нарушения, дата и время, тип маневра, степень опасности маневра, количество штрафных баллов.

В результате проведенного выше анализа была спроектирована архитектура приложения, представленная на рис 4.

Таблица 2

Пороговые значения ускорения легкового автомобиля

Скорость(км/ч)	**Ускорение(м/с^2)**	**Маневр**
<20	2.4–2.9	Резкий
	3.0–3.4	Опасный
	>3.4	Критичный
20–60	3.0–4.0	Резкий
	4.1–5.0	Опасный
	>5.0	Критичный
>60	4.0–5.0	Резкий
	5.1–5.6	Опасный
	>5.7	Критичный

Таблица 3

Пороговые значения торможения легкового автомобиля

Скорость(км/ч)	**Ускорение(м/с^2)**	**Маневр**
<20	2.9–3.4	Резкий
	3.5–4.0	Опасный
	>4.1	Критичный
20–60	3.2–4.0	Резкий
	4.1–4.8	Опасный
	>4.9	Критичный
>60	4.4–5.2	Резкий
	5.3–5.8	Опасный
	>5.9	Критичный

Таблица 4

Пороговые значения поворота легкового автомобиля

Скорость(км/ч)	**Ускорение(м/с^2)**	**Маневр**
<20	>90	Резкий
20–40	45–90	Резкий
	>90	Опасный
40–60	23–45	Резкий
	46–90	Опасный
	>90	Критичный
>60	23–45	Опасный
	>45	Критичный

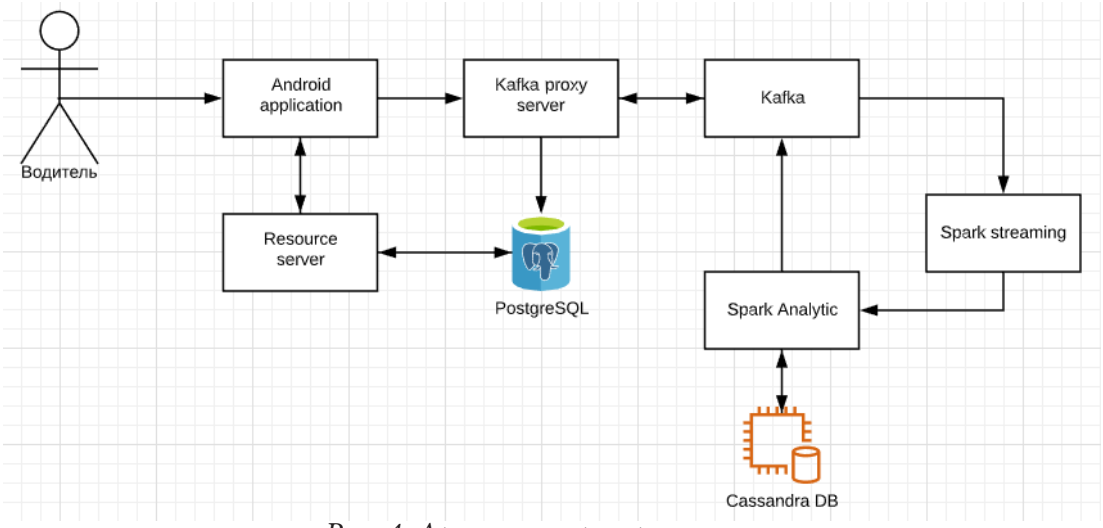


Рис. 4. Архитектура приложения

Заключение

В данной работе был приведен алгоритм анализа качества вождения ТС. Был разработан алгоритм получения исходных данных с датчиков устройств, их обработка, разработаны модули на Apache Spark для обработки больших данных с сохранением результата в базу данных.

Литература

1. Комплементарный фильтр. – Режим доступа: https://robotclass.ru/articles/complementary-filter. – (Дата обращения: 14.01.2020).

2. Фильтр Маджвика. – Режим доступа: https://habr.com/ru/post/255661. – (Дата обращения: 14.01.2020).

3. Фильтр Калмана. – Режим доступа: https://habr.com/ru/post/166693. – (Дата обращения: 14.01.2020).

4. Изучаем Spark: молниеносный анализ данных : книга / Х. Карау [и др.]; под ред. Х. Карау. – Москва : ДМК Пресс, 2015. – 304 с.

Тарасова Виктория Олеговна – магистрант 2-го года обучения кафедры МО ЭВМ Воронежского государственного университета. E-mail: vika-tarasova-1996@mail.ru

Трофименко Елена Владимировна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры МО ЭВМ Воронежского государственного университета. E-mail: evtrof@gmail.com

УДК 519.72

ОБ ОДНОМ ПОДХОДЕ К ВЫБОРУ ОПТИМАЛЬНОГО РАЗБИЕНИЯ

Е. А. Тихомирова

Воронежский государственный университет

Введение

В настоящее время существует значительное количество методов кластеризации [1], среди которых определенный интерес представляют методы, которые ориентированы на обработку приближенной информации, а также разнородных данных [2]. Исходная информация для таких методов формируется в виде матрицы нечеткого отношения сходства/несходства. Ее обработка в соответствии с некоторой схемой позволяет определить альтернативные варианты кластеризации, из которых выбирается подходящий с привлечением дополнительных критериев. В [3–5] рассматривается *метод декомпозиционного дерева*, идея которого первоначально была изложена в [6]. Данный метод позволяет построить классы объектов, расстояние между которыми не превосходит задаваемой в качестве порога величины уровня, при этом в качестве расстояний выступают транзитивные расстояния [6].

Метод декомпозиционного дерева обладает рядом преимуществ:

– возможность работать с разнородной информацией за счет того, что формируемое на начальном шаге отношение несходства базируется на специальных функциях (индексы несходства, функция подобия и др.), область определения которых может быть произвольной, связанной со шкалой измерений, а область значений всегда можно представить в виде [0,1];

– декомпозиционное дерево отражает последовательный процесс разбиения заданного множества объектов на классы, причем для каждого объекта можно проследить его эволюцию от ситуации, когда все объекты попадают в один класс, до ситуации, когда данный объект образует уникальный класс;

– использование параметрических нечетких операций (в частности, треугольных норм и конорм) позволяет осуществить настройку метода на конкретного пользователя или конкретную проблему.

Одним из основных преимуществ этого метода является возможность получения всех потенциальных разбиений исходного множества. Поэтому возникает проблема выбора наиболее подходящего разбиения, что обусловливает необходимость разработки метода выбора *оптимального* в некотором смысле разбиения и его оценки.

Цель статьи заключается в представлении подхода к выбору наиболее подходящей кластерной структуры на основе введенных показателей качества разбиения, при этом система показателей не является жесткой и может быть дополнена другими показателями, уточняющими понятие «оптимальности» в зависимости от специфики конкретной задачи.

1. Метод декомпозиционного дерева

Рассмотрим задачу нечеткой кластеризации. Пусть задано множество объектов X, причем каждый объект характеризуется набором признаков $P_1,...,P_m$, так что каждому объекту $x_i \in X\ (i=1,n)$ ставится в соответствие векторная оценка $v_i = \left(v_i^1,...,v_i^m\right)$. Требуется определить классы ближайших объектов.

Как правило, «близость» оценивается с помощью подходящей функции расстояния. В методе декомпозиционного дерева используются так называемые транзитивные расстояния. Матрица транзитивных расстояний совпадает с матрицей отношения различия, которое формируется путем транзитивного замыкания отношения несходства. Заметим, что дополнением к отношению различия является отношение эквивалентности. Именно отношение эквивалентности лежит в основе любой задачи кластеризации. На рис. 1 представлена схема метода декомпозиционного дерева. Для получения альтернативных разбиений заданного множества возможно использование различных вариантов метода.

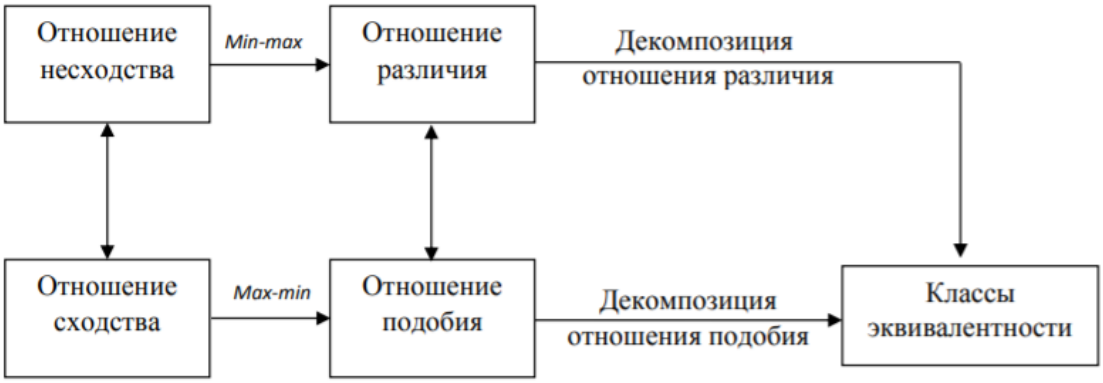


Рис. 1. Схема метода декомпозиционного дерева

На основе исходного множества строится отношение сходства или несходства с помощью функции расстояния. Для перехода от отношения несходства к отношению сходства или наоборот используется операция дополнения. Заметим, что отношения сходства и несходства не обладают свойством транзитивности. Для обеспечения этого свойства используется операция транзитивного замыкания. Полученное отношение называется отношением подобия (различия), оно обладает свойствами: рефлексивности (антирефлексивности), симметричности и транзитивности. Согласно теореме декомпозиции любой α-срез подобия является обычным отношением эквивалентности и, следовательно, определяет разбиение множества объектов на непересекающиеся классы эквивалентности. Наглядно работу метода можно представить в виде схемы, которая называется *декомпозиционным деревом*.

2. Выбор оптимального разбиения в методе декомпозиционного дерева

Пусть на множестве объектов X сформировано отношение подобия $R \subseteq X \times X$, которому соответствует матрица $[R] = (r_{ij})_{n \times n}$, где $n = |X|$ – количество объектов во множестве X, $r_{ij} = \mu_R(x_i, x_j) \in [0,1]$ — степень, с которой объекты x_i и x_j можно считать эквивалентными, равноценными. Отношение подобия R – это нечеткое отношение, поэтому его представить по теореме декомпозиции [6] в виде

$$R = \bigcup_{\alpha} (\alpha \cdot R_\alpha),$$

где $\alpha \in (0,1]$ – значение параметра, R_α – α-срез нечеткого отношения R, которое является обычным отношением эквивалентности.

Известно, что каждому отношению эквивалентности, соответствует совокупность классов эквивалентности, образующих фактор-множество. Перебирая возможные значения α, каждый раз будем получать «свое» разбиение. Упорядоченная последовательность значений параметра α формирует цепочку включений для классов эквивалентности, т. е. при увеличении α классы эквивалентности могут только разбиваться на более мелкие. Таким образом, в резуль-

тате применения метода формируется декомпозиционное дерево, отображающее процесс последовательного разбиения классов. Нижний уровень дерева соответствует ситуации, когда множество объектов X образует единый класс. Верхний уровень содержит хотя бы один тривиальный класс – такой, который состоит из единственного объекта.

Таким образом, декомпозиционное дерево включает множество разбиений, каждое из которых, в свою очередь, состоит из различных классов. Количество ярусов декомпозиционного дерева зависит от того, каким образом формируется множество значений параметра α: или это множество несовпадающих значений элементов матрицы отношения подобия (1), или множество узловых точек, образующихся при разбиении промежутка $[0,1]$ на k равных частей (2). Наша задача заключается в том, чтобы выбрать некоторое разбиение, т. е. ярус декомпозиционного дерева, который по сравнению с другими является более предпочтительным. Очевидно, что для решения данной задачи необходимо привлекать дополнительные критерии. Поскольку решается задача кластеризации, то можно рассматривать классические критерии качества, такие как *компактность* и *отделимость* [1].

Пусть $\{\alpha_i\}$ — множество значений параметра α, которое сформировано по способу (1), а $\{\widehat{\alpha}_j\}$ — множество значений параметра α, которое сформировано по способу (2) с постоянным шагом h, т. е. $\widehat{\alpha}_{j+1} = \widehat{\alpha}_j + h$. Во избежание выбора тривиального разбиения исключим из рассмотрения соответствующие $\widehat{\alpha}_j$-уровни, кроме последнего.

Каждому α_i-уровню (кроме последнего) поставим в соответствие величину

$$\theta_i = \frac{\alpha_i - \alpha_{i-1}}{h}. \tag{1}$$

Данная величина показывает количество ярусов, на которых разбиение, соответствующее α_i, не изменится. Каждому кластеру Cl_k, который представлен в декомпозиционном дереве, поставим в соответствие отрезок $\left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]$ —промежуток значений параметра α, которым соответствуют ярусы декомпозиционного дерева, содержащие кластер Cl_k, а также значение величины

$$\gamma_k = \sum_{\alpha_i \in \left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]} \theta_i \,. \tag{2}$$

Отсортируем полученные данные по невозрастанию величины γ_k. Затем, двигаясь по списку, будем выбирать подходящие кластеры таким образом, чтобы отрезок $\left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]$ сокращался только слева или только справа, дабы один и тот же объект не мог попасть в разные кластеры.

В качестве оценки качества кластеризации будем использовать коэффициент силуэта [1]

$$Sil_\alpha = \frac{1}{n} \sum_{x_i \in X} \frac{b_i - a_i}{\max(a_i, b_i)} \in [-1,1], \tag{3}$$

где a_i – среднее расстояние от x_i до объектов того же кластера, b_i – среднее расстояние от x_i до объектов из других кластеров.

Значения, близкие к 1, соответствуют плохим (неправильным) кластеризациям. Значения, близкие к 0, свидетельствуют о том, что кластеры пересекаются и накладываются друг на друга. Значения, близкие к 1, соответствуют плотно сгруппированным кластерам.

Пусть K_α – разбиение, которое соответствует уровню α. Для характеристики относительного размера кластеров, входящих в разбиение, введем следующий показатель

$$WCL_\alpha = 1 - \frac{MaxCl_\alpha - MinCl_\alpha}{n-2}, \tag{4}$$

где $MaxCl_\alpha$ – количество объектов в максимальном кластере; $MinCl_\alpha$ – количество объектов в минимальном кластере.

Можно заметить, что если объекты распределены по кластерам равномерно, то $MaxCl_\alpha = MinCl_\alpha$ и тогда $WCL_\alpha = 1$. При наибольшем различии в размерах кластеров $WCL_\alpha = 0$.

Для оценки близости разбиения к тривиальному введем другой показатель

$$DistTriv_\alpha = \frac{TrivCl_\alpha}{n}, \tag{5}$$

где $TrivCl_\alpha$ – количество тривиальных кластеров.

3. Иллюстративный пример

На рис. 2 представлено исходное множество объектов в виде множества точек на плоскости. Матрица отношения несходства – это матрица расстояний между этими точками.

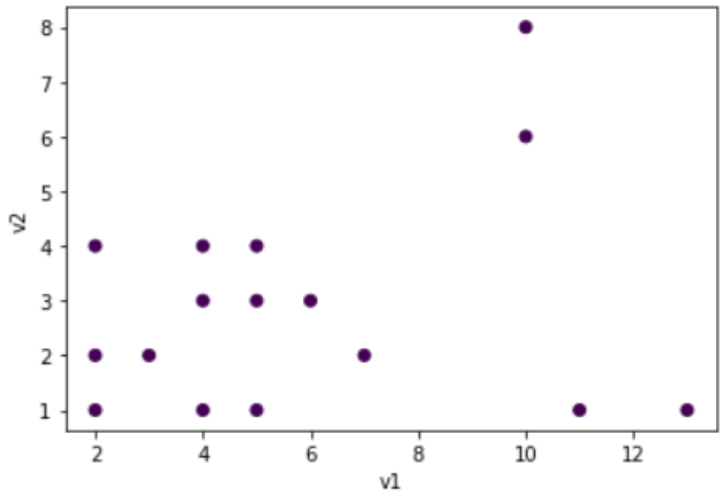


Рис. 2. Исходное множество X

На основе минмаксного транзитивного замыкания было сформировано отношение различия, его дополнением является отношение подобие. Каждый α -срез данного отношения является эквивалентностью и индуцирует разбиение на классы эквивалентности. На рис. 3 представлено соответствующее декомпозиционное дерево. Как можно заметить, с уменьшением α происходит укрупнение классов эквивалентности. Слева указано количество уровней $\hat{\alpha}_j$, совпадающих с уровнем α_i.

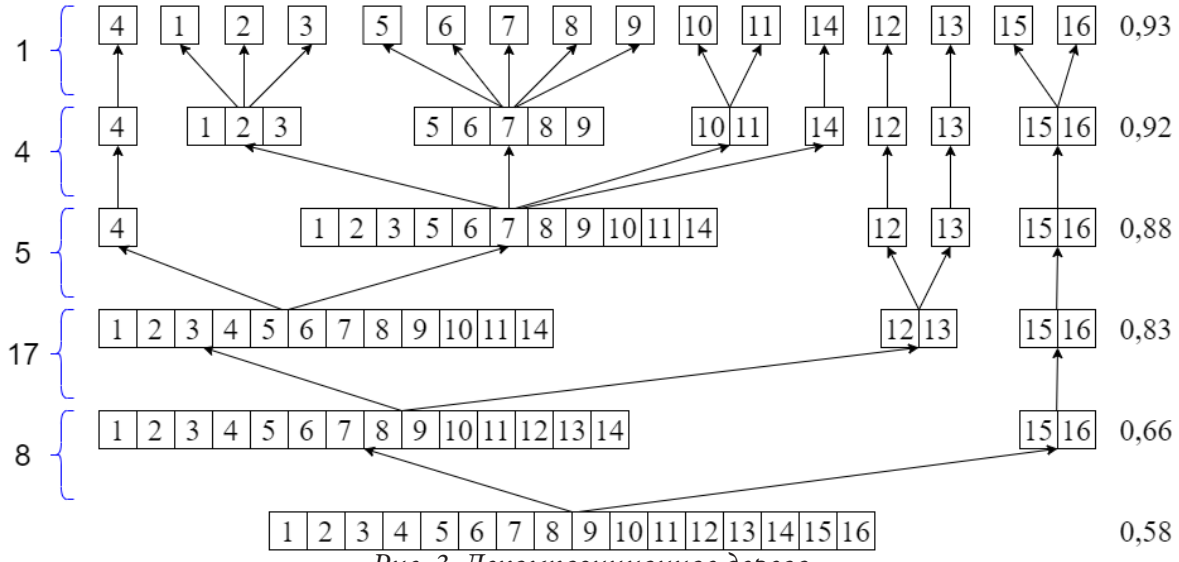


Рис. 3. Декомпозиционное дерево

В табл. 1 приведены оценки θ_k для каждого α-уровня, вычисленные по формуле (1).

Таблица 1

θ-оценки

α	0, 93	0, 92	0, 88	0, 83	0, 66
θ	1	4	5	17	8

Построим таблицу, в которой перечислены значения следующих показателей: A_i – множество уровней декомпозиционного дерева, на которых встречается кластер Cl_j; θ_j – оценка для j-го кластера, вычисленная по формуле (2).

Таблица 2

Характеристики кластеров

№	Cl_j	A_i	θ_j
1	{1, 2, 3}	{0,92}	4
2	{4}, {12}, {13}	{0,93; 0,92; 0,88}	10
3	{5, 6, 7, 8, 9}	{0,92}	4
4	{10, 11}	{0,92}	4
5	{14}	{0,93; 0,92}	5
6	{15, 16}	{0,92; 0,88; 0,83; 0,66}	34
7	{1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 14}	{0,88}	5
8	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 14}	{0,83}	17
9	{12, 13}	{0,83}	17
10	{1}, {2}, {3}, {5}, {6}, {7}, {8}, {9}, {10}, {11}, {14}, {15}, {16}	{0,93}	1
11	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14}	{0,66}	8

Отсортируем табл. 2 по невозрастанию показателя γ_j и каждому кластеру Cl_j поставим в соответствие промежуток $\left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]$. Полученные данные приведены в табл. 3.

Таблица 3

Выбор оптимального разбиения

№	Cl_j	$\left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]$	γ_j
1	{15, 16}	[0,59; 0,92]	34
2	{12, 13}	[0,67; 0,83]	17
3	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 14}	[0,67; 0,83]	17
4	{4}, {12}, {13}	[0,84; 0,93]	10
5	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14}	[0,59; 0,81]	8
6	{14}	[0,89; 0,93]	5
7	{1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 14}	[0,84; 0,88]	5
8	{1, 2, 3}	[0,89; 0,92]	4
9	{10, 11}	[0,89; 0,92]	4
10	{5, 6, 7, 8, 9}	[0,89; 0,92]	4
11	{1}, {2}, {3}, {5}, {6}, {7}, {8}, {9}, {10}, {11}, {14}, {15}, {16}	[0,93; 0,93]	1

Двигаясь по таблице сверху вниз, будем выбирать кластеры, содержащие элементы, которые еще не были распределены. Помним, что отрезок $\left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]$ может только сужаться. Кластер {15,16} сохраняется на 34 уровнях, а кластеры {1,2,3,4,5,6,7,8,9,10,11,14}, {12,13} на 17, при этом промежуток $\left[\underline{\alpha}(Cl_k), \overline{\alpha}(Cl_k)\right]$ изменяется следующим образом:

$$[0.59, 0.92] \rightarrow [0.67, 0.83] \rightarrow [0.67, 0.83].$$

В полученное разбиение попали все объекты из X, при этом найденный промежуток [0.67,0.83] соответствует $\alpha = 0.83$. Результат кластеризации приведен на рис. 4.

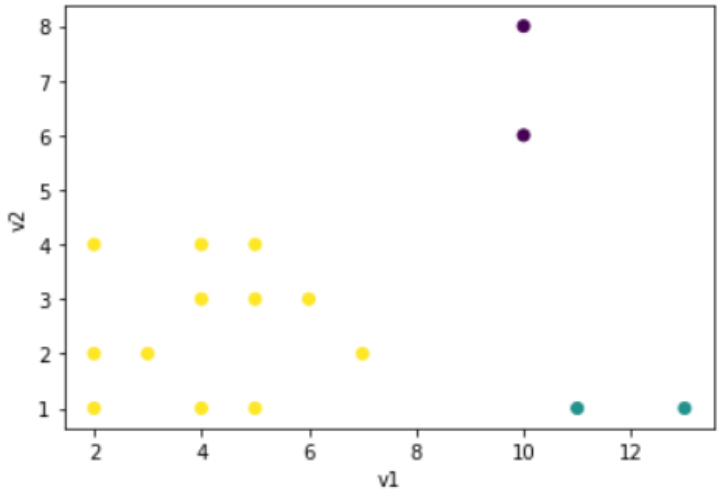


Рис. 4. Классы разбиения объектов из множества X

Для каждого разбиения, которое соответствует ярусу декомпозиционного дерева, вычислены оценки (3), (4) и (5) (табл. 4). Нетрудно заметить, что выбранная кластерная структура $K_{0.83}$ обладает наибольшим значением коэффициента силуэта $Sil_{0.83} = 0.65$ среди других разбиений.

Таким образом, данный подход позволяет определить значение уровня α, то есть определить разбиение исходного множества объектов на наиболее плотно сгруппированные кластеры.

Таблица 4

Оценки разбиений

α	Sil_α	WCl_α	$DistTriv_\alpha$
0,92	0,33	0,85	0,25
0,88	0,18	0,35	0,19
0,83	0,65	0,3	0
0,66	0,47	0,14	0

Работа выполнена под руководством доктора техн. наук, профессора Леденевой Т. М.

Литература

1. Лекции по метрическим алгоритмам классификации. – Режим доступа: http:// www.ccas.ru/frc/papers/voron04mpc.pdf. (Дата обращения: 4.03.2020).

2. *Billard, L.* Symbolic Data Analysis: Conceptual statistics and Data Mining / L. Billard, E. Diday. – John Wiley. & Sons Ltd, 2006. – 345 pp.

3. *Леденева, Т. М.* Обработка нечеткой информации / Т. М. Леденева. Воронеж : Изд-во ВГУ, 2006. – 233 с.

4. *Каплиева, Н. А.* Исследование различных типов транзитивности в приложении к нечеткой классификации / Н. А. Каплиева, Т. М. Леденева // Вестник ВГУ. Серия: Физика. Математика, 2006. – № 2. – С. 206–216.

5. *Леденева, Т. М.* О влиянии функции подобия на результаты нечеткой классификации / Т. М. Леденева, Нгуен Нгок Хуи // Информационные технологии. – М. : Новые технологии, 2011. – № 11. – С. 15–23.

6. *Кофман, А.* Введение в теорию нечетких множеств / А. Кофман. – М. : Радио и связь, 1982. – 432 с.

Тихомирова Екатерина Александровна – студентка 4-го курса кафедры Вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: katya246893@gmail.com

Леденёва Татьяна Михайловна (научный руководитель) – д-р техн. наук, проф., зав. кафедрой вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: tm-ledeneva@yandex.ru

УДК 519.6

ПРИМЕНЕНИЕ АЛГОРИТМОВ ВЫДЕЛЕНИЯ КОНТУРОВ В ЗАДАЧЕ МЕТАЛЛОГРАФИИ

Е. В. Ткач

Воронежский государственный университет

Введение

Количественная металлография – это совокупность методов количественной оценки геометрических параметров пространственного строения металлов и сплавов.

Простейшим видом количественного анализа является визуальная оценка структуры – «мельче, крупнее, однородное или нет и насколько». В современной металлографии этого недостаточно. Необходимы точные количественные оценки для того, чтобы проследить кинетику изменения структуры в процессе внешнего воздействия (термического, механического и т. д.) и определить механизмы реализующихся процессов.

Стандартный подход предполагает переход от общего к частному – создание эталонов структур (которые имеют количественную оценку), которыми можно пользоваться для количественной оценки структур изучаемых материалов [6].

Определение площади объектов в программе обработки изображений – это наиболее объективный вид анализа. Для подсчета площадей объектов необходимо сначала выделить контуры на изображении. Выделение границ объектов одна из сложных задач в обработке изображений, особенно в случае сильно зашумленного изображения.

1. Постановка задачи

В данной работе рассматривается задача закраски близких по площади областей на изображении шлифа металла. Для решения поставленной задачи необходимо качественно выделить контур. На рис. 1 представлены исходное и итоговое изображение, которое должно быть результатом работы программы.

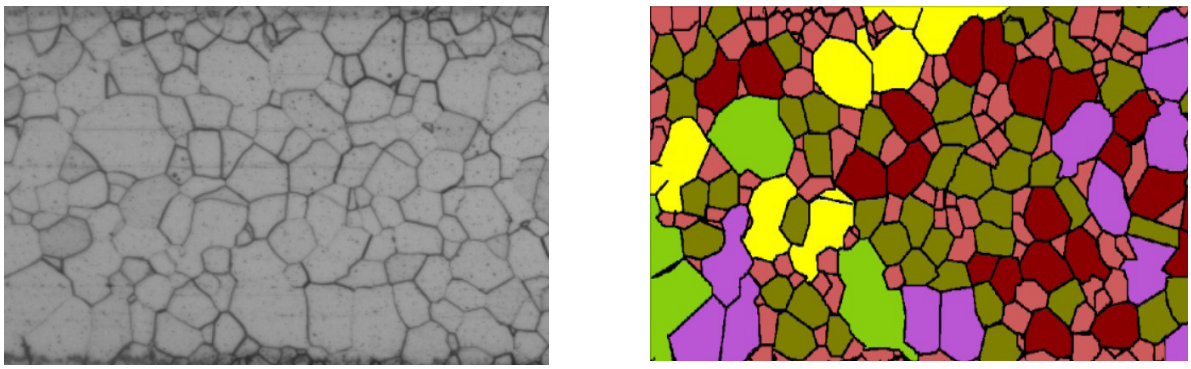


Рис. 1. Примеры исходного и результирующего изображений

Подобная задача может быть решена методами машинного обучения [7], но в данной работе предполагается попытка ее решения методами компьютерной графики.

Разработан алгоритм, который включает следующие шаги:

1. Перевод исходного изображения в двоичное.
2. Применение алгоритма водораздела для поиска контуров.

3. Постобработка с применением морфологических преобразований.

4. Подсчет площадей выделенных зерен.

5. Закраска близких по площади областей.

В процессе применения стандартных алгоритмов и методов выделения контуров возникают трудности: появляются ложные контуры, разрывы контура. Поэтому в данной задаче обнаружить границы предлагается с помощью алгоритма водораздела [1–4].

Сегментация методом водораздела подразумевает рассмотрение изображения в качестве некоторой карты местности, где значения яркостей представляют собой значения высот относительно некоторого уровня. Если начать эту местность заполнять водой, то получим затопленные области, так называемые бассейны. При дальнейшем заполнении водой эти бассейны объединяются, т. е. маленькие бассейны постепенно сливаются вместе в большие бассейны. Места объединения этих бассейнов отмечаются как линии водораздела. Области формируются с помощью локальной геометрической структуры, чтобы связать особенности различных областей изображения с полученными измеренными локальными экстремумами.

2. Результаты исследования

Для тестирования разработанного алгоритма было создано эталонное изображение (рис. 2) с четким контуром областей на основе исходного изображения (ручная разметка границ).

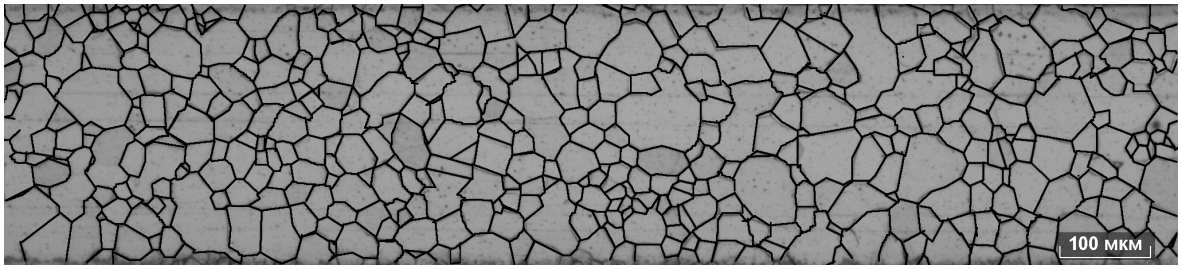


Рис. 2. Эталонное изображение

На рис. 3 показан первый этап алгоритма, а именно – перевод исходного изображения в двоичное (рис. 3). Бинаризация необходима для того, чтобы разделить изображение на области, для которых выполняется определенный критерий однородности. Понятие области изображения используется для определения связной группы элементов изображения, имеющих общий признак. Порог – признак, который помогает разделить искомый сигнал на классы. Бинаризация заключается в сопоставлении значения яркости каждого пикселя изображения.

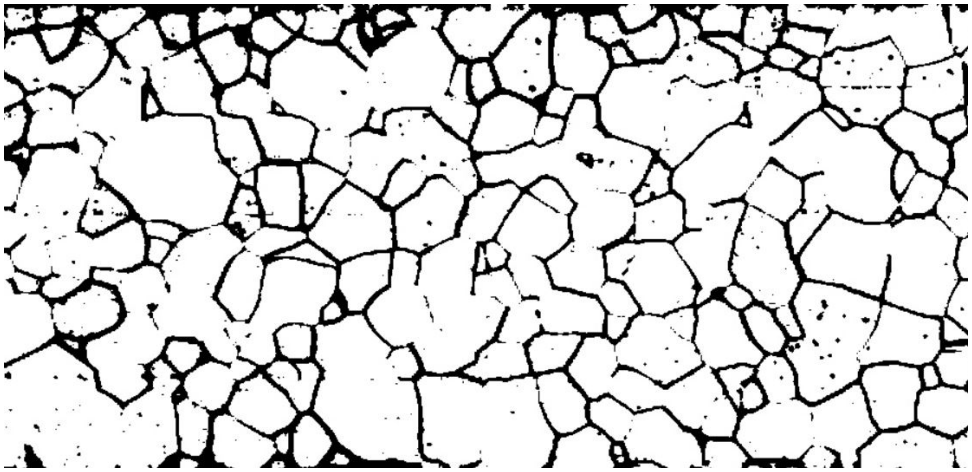


Рис. 3. Исходное изображение в двоичном виде

После применения алгоритма водораздела (рис. 4) для бинарного изображения (рис. 3) можно заметить, что качество контура улучшилось, водораздел достроил недостающий контур. Но есть и минусы данного алгоритма – контур в некоторых местах стал значительно толще.

Основной идеей корректировки результата выделения контура является использование методов морфологических преобразований: дилатации и эрозии [5]. Дилатация (морфологическое расширение) – свертка изображения или выделенной области изображения с некоторым ядром. Такая операция вызывает рост светлых областей на изображении. Эрозия (морфологическое сужение) – обратная операция. К изображению, полученному применением алгоритма водораздела, была применена дилатация, в результате чего контур был немного сужен (рис. 5).

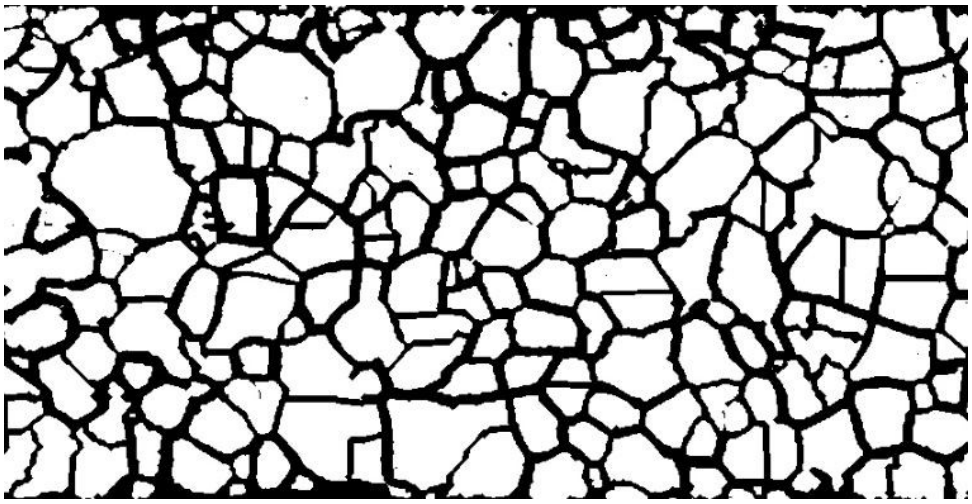


Рис. 4. Применение водораздела к изображению на рис. 3

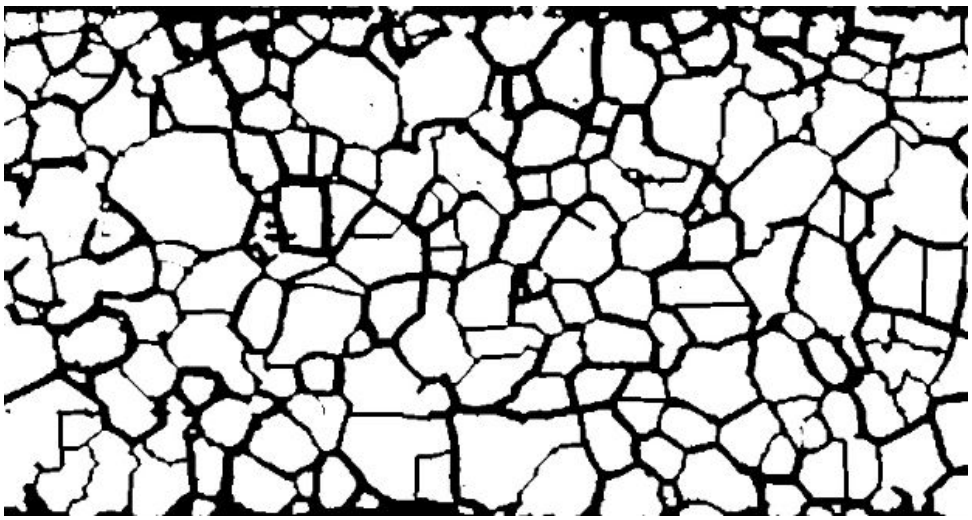


Рис. 5. Применение дилатации к изображению на рис. 4

На рис. 6, 7 представлены результаты закраски близких по площади областей для изображения на рис. 5 и для эталонного изображения.

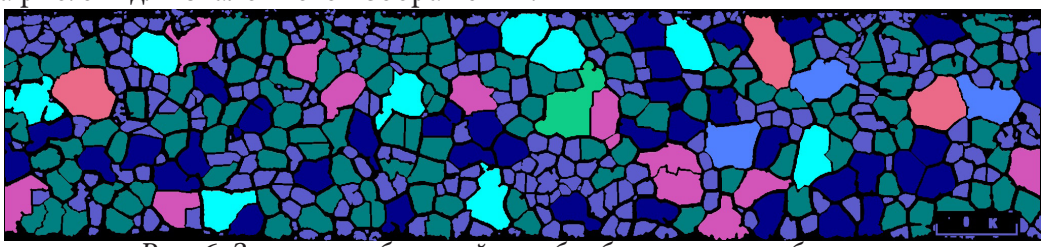


Рис. 6. Закраска областей на обработанном изображении

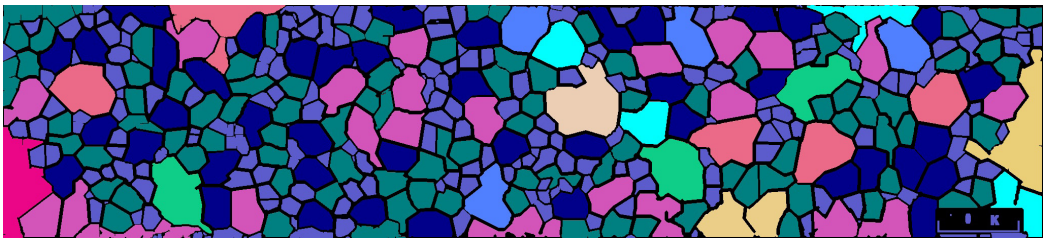


Рис. 7. Закраска областей на эталонном изображении

Оценка качества работы алгоритма была проведена путем вычитания выделенных контуров на исходном и эталонном изображении (рис. 8). Белые штрихи означают, что данный контур не совпадает на исходном и эталонном изображении.

Рис. 8. Результат вычитания контуров эталонного и исходного изображения

Заключение

На данном этапе работы удалось реализовать алгоритмы водораздела, морфологии и закраски близких по площади сегментов, а также сравнить результаты работы алгоритма на исходном и эталонном изображении.

Рассмотрены различные варианты комбинирования алгоритмов выделения контуров и методов математической морфологии.

Основной проблемой остаются ложные контуры или же потери контуров, которые не могут быть компенсированы морфологическими преобразованиями и другими методами постобработки и предобработки. Также при попытке улучшении качества выделенного контура значительно увеличивается его толщина, что влечет за собой не совсем корректный подсчет площадей.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н. и канд. физ.-мат. наук Медведевой О. А.

Литература

1. *Анисимов, Б. В.* Распознавание и цифровая обработка изображений: Учебное пособие для студентов вузов / Б. В. Анисимов, В. Д. Курганов, В. К. Злобин. – Москва : Высшая школа, 1983. – 295 с.

2. *Бондина, Н. Н.* Использование статистических характеристик для выделения границ в медицинских изображениях / Н. Н. Бондина, В. Э. Кривенцов // Вестник НТУ ХПИ. Серия : Информатика и моделирование. – Харьков : НТУ ХПИ. – 2013. – № 39 (1012). – С. 22–27.

3. *Гонсалес, Р.* Цифровая обработка изображений / Р. Гонсалес, Р. Вудс ; пер. с англ. П. А. Чочиа. – Москва :Техносфера, 2012. – 1072 с.

4. *Фисенко, В. Т.* Компьютерная обработка и распознавание изображений / В. Т. Фисенко, Т. Ю. Фисенко. – СПб. : СПбГУ ИТМО, 2008. – 192 с.

5. *Визильтер, Ю. В.* Структурная фильтрация цифровых изображений с использованием проективных морфологий / Ю. В. Визильтер // Вестник компьютерных и информационных технологий. – 2008. – № 3. – С. 18–22.

6. Количественный анализ изображений. – Режим доступа: http://structure.by/index.php

7. *Ковун, В. А.* Разработка моделей и алгоритмов машинного обучения для автоматического металлографического определения наблюдаемых размеров зерен стали / В. А. Ковун, И. Л. Каширина // Актуальные проблемы прикладной математики, информатики и механики: сб. тр. междунар. конф., Воронеж, 11-13 ноября 2019 г. – С. 249–255.

Ткач Екатерина Владиславовна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: yekaterina.tkatch2016@yandex.ru

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

Медведева Ольга Александровна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 004.94

АНАЛИЗ АЛГОРИТМОВ ОБРАБОТКИ КОЛЛИЗИЙ В МЕТОДЕ PBF НА ОСНОВЕ КАРТ ПЛОТНОСТЕЙ

М. И. Токарев, Е. В. Трофименко

Воронежский государственный университет

Введение

В области компьютерной графики существуют задачи, где требуется смоделировать и визуализировать поведение жидкости в тех или иных условиях. Чаще всего для этого используется подход с использованием гидродинамики сглаженных частиц с различными модификациями [1]. Одной из таких модификаций является алгоритм PBF (Position Based Fluids), который был предложен в 2013 году [2]. Данный алгоритм превосходит по скорости вариации SPH алгоритма (PCISPH, IISPH, DFSPH), но при этом уступает в точности, что делает его подходящим только для областей, где не требуется точный расчет (игры, спецэффекты). Как и любой другой алгоритм симуляции жидкости, PBF требует правильно подобранного метода обработки коллизий. В настоящей статье описываются основные методы обработки коллизий, а также даются рекомендации по их применению.

1. Основные уравнения

Жидкость можно рассматривать как совокупность движущихся частиц, где каждая частица с номером i обладает собственной массой m_i давлением p_i и плотностью ρ_i. Частица $\mathbf{x}_i$ движется со скоростью $\mathbf{v}_i$:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{v}_i. \tag{1}$$

Для описания движения жидкости используется уравнение Навье – Стокса [3]:

$$\frac{d\mathbf{v}_i}{dt} = -\frac{1}{\rho_i}\nabla p_i + \nu \nabla \mathbf{v}_i^2 + \frac{\mathbf{F}_i}{m_i}, \tag{2}$$

где ν — коэффициент кинематической вязкости, а $\mathbf{F}_i$ — сумма внешних сил, действующих на частицу, движущуюся в потоке. Член уравнения (2) $-\frac{1}{\rho_i}\nabla p_i$ описывает ускорение частицы под действием разностей давления окружающих частиц. $\nu\nabla\mathbf{v}_i^2$ представляет собой ускорение под действием силы трения между частицами с разными скоростями. Ускорение от воздействия внешних сил описывается как $\frac{\mathbf{F}_i}{m_i}$.

2. Описание метода

PBF является логическим дополнением PBD (Position-Based Dynamics) подхода и дает хорошую производительность [4]. Его псевдокод приведен в листинге 1. Идея метода PBF состоит в том, что на каждую частицу с позицией $\mathbf{x}_i$ накладывается ограничение на плотность частицы ρ_i в следующем виде:

$$C_i(\mathbf{x}_1,...,\mathbf{x}_n) = \frac{\rho_i}{\rho_0} - 1, \tag{3}$$

где ρ_0 – это исходная плотность, а ρ_i вычисляется с использованием гидродинамики сглаженных частиц (англ. Smooth Particle Hydrodynamics):

$$\rho_i = \sum_j m_j W(\| \mathbf{x}_i - \mathbf{x}_j \|, h)$$
$$W(q,h) = \frac{f(q)}{h^3}. \tag{4}$$

В качестве функции ядра часто используются функцию нормального распределения или кубический сплайн, который равен нулю для всех частиц, находящихся дальше, чем $2h$, что позволяет сэкономить вычислительные ресурсы [1].

Обозначим $\mathbf{x}_1,...,\mathbf{x}_n = \mathbf{x}$. Цель PBF найти такое $\Delta\mathbf{p}$, что выполняется условие:

$$C(\mathbf{x} + \Delta\mathbf{x}) = 0. \tag{5}$$

В алгоритме 1 приведен псевдокод всего метода PBF.

Листинг 1

Псеводкод метода PBF

for all частица i do
    вычислить скорость на основе действующих внешних сил $\mathbf{v}_i := \mathbf{v}_i + \Delta t \mathbf{f}_e xt(\mathbf{x}_i)$
    предсказать следующую позицию на основе действующих внешних сил $\mathbf{x}_i^* := \mathbf{x}_i + \Delta t \mathbf{v}_i$
end for
for all частица i **do**
    найти ближайшие частицы j
end for
while iter < solverIteration **do**
    for all частица i **do**
        вычислить поправку позиции $\Delta\mathbf{x}_i$
        обработать коллизии и применить правки к $\Delta\mathbf{x}_i$
    end for
    for all частица i **do**
        скорректировать предсказанную ранее позицию на основе правки $\mathbf{x}_i^* := \mathbf{x}_i^* + \Delta\mathbf{x}_i$
    end for
end while
for all частица i **do**
    обновить значение скорости $\mathbf{v}_i := \frac{1}{\Delta t}(\mathbf{x}_i^* - \mathbf{x}_i)$
    вычислить окончательную позицию частицы $\mathbf{x}_i := \mathbf{x}_i^*$
end for

3. Обработка коллизий

Подходы к обработке коллизий на основе плотности заключаются в том, что на частицу действуют другие частицы, а также твердые тела в области симуляции, которые обладают своей собственной плотностью. Таким образом, в случае если частица находится рядом с твердым телом, то при вычислении градиента C_i по формуле 3 частица приобретет большую корректировку к позиции и тем самым коллизии не произойдет.

Для того чтобы вычислить плотность окружающих тел, которые влияют на частицу требуется дискретизировать область симуляции. Существует два основных способа как это сделать:

1. Представить тело как множество частиц с очень большим значением плотности (рис. 1) [5].

2. Дискретизировать пространство в виде сетки с равным шагом и перед началом симуляции просчитать значение плотности твердого тела в каждой точке по формуле 6 (рис. 2) [6].

$$\rho_B(\mathbf{x}) = \int_{\Omega_B} \Phi(\mathbf{x}^*, h) W(\| \mathbf{x} - \mathbf{x}^* \|, h) d\mathbf{x}^*$$
$$\Phi(\mathbf{x}) = s(\mathbf{x}) \inf_{\mathbf{x}^* \in \partial B} \| \mathbf{x} - \mathbf{x}^* \|, \quad (6)$$

где Ω_B – область влияния твердого тела на частицу, $s(\mathbf{x})$ – функция, возвращающая –1, если $\mathbf{x}$ внутри твердого тела и 1 иначе.

Для того чтобы определить в какой ситуации использовать тот или иной подход, были реализованы CUDA версии обоих алгоритмов и проведено их сравнение на основе производительности на машине со следующими характеристиками:

- Видеокарта NVIDIA GTX 1070
- Процессор AMD Ryzen 5 1600
- Оперативная память 32 ГБ

Область симуляции содержит 1 млн частиц. Полученные результаты приведены в табл. 1.

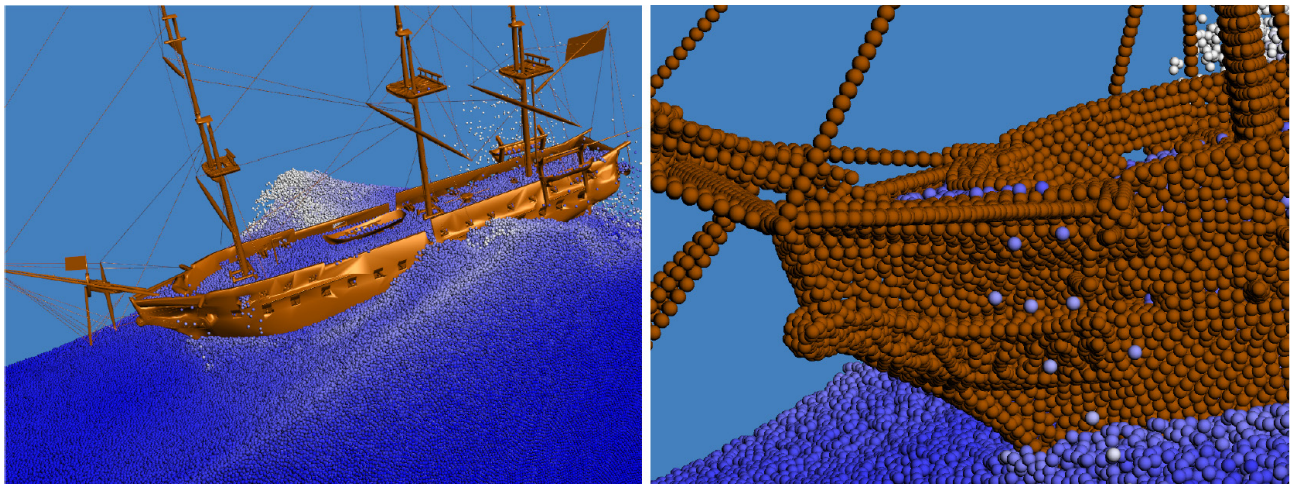


Рис. 1. Дискретизация твердого тела частицами

Рис. 2. Дискретизация сеткой. Карта, построенная на основе плотности твердого тела в каждой ячейке сетки

Таблица 1

Среднее время выполнения этапа обработки коллизий на каждой итерации

Этап симуляции	Дискретизация частицами, мс	Дискретизация сеткой, мс
Поиск соседей	153	127
Вычисление правки $\Delta \mathbf{x}_i$	79	68
Всего	232	195

Заключение

Данные из табл. 1 показывают, что дискретизация сеткой дает наилучшие результаты, однако данный способ имеет свои недостатки. Так, например, если в области симуляции присутствует мягкое тело, то карту плотностей нужно пересчитывать на каждой итерации, что резко снизит общую скорость вычислений. Также стоит отметить сложность реализации такого подхода по сравнению с дискретизацией частицами. Несмотря на эти недостатки данный подход является весьма перспективным и в настоящее время ведутся исследования по улучшению данного алгоритма, которые позволят избавиться от этих недостатков. Таким образом, на данный момент можно сделать вывод, что если в сцене отсутствуют тела, требующие пересчета карты плотностей, то предпочтительней является метод дискретизации сеткой.

Литература

1. Гидродинамика сглаженных частиц. – Режим доступа: https://ru.wikipedia.org/wiki/Гидродинамика_сглаженных_частиц (Дата обращения: 18.01.2020).

2. *Macklin, M.* Position Based Fluids / M. Macklin, M. Müller // ACM Transaction On Graphics. – 2013. – P. 32–36.

3. Уравнение Навье – Стокса. – Режим доступа: https://ru.wikipedia.org/wiki/Уравнение_Навье_—_Стокса (Дата обращения: 18.01.2020).

4. *Müller, M.* Position Based Dynamics / M. Müller, B. Heidelberger, M. Hennix, J. Ratcliff // Journal of Visual Communication and Image Representation. – 2007 – P. 109–118.

5. *Akinci, N.* Versatile rigid-fluid coupling for incompressible SPH / N. Akinci, M. Ihmsen, G. Akinci, B. Solenthaler, M. Teschner // ACM Transaction on Graphics. – 2012 – P. 31–39.

6. *Koschier, D.* Density Maps for Improved SPH Boundary Handling / D. Koschier, J. Bender // ACM SIGGRAPH/Eurographics Symposium on Computer Animation. – 2017.

Токарев Максим Игоревич – магистрант 2-го года обучения кафедры МО ЭВМ Воронежского государственного университета. E-mail: tokarevmaxim.stud@gmail.com

Трофименко Елена Владимировна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры МО ЭВМ Воронежского государственного университета. E-mail: evtrof@gmail.com

УДК 004.4, 004.7

ОСОБЕННОСТИ ТЕХНОЛОГИИ SPRING FRAMEWORK В РАЗРАБОТКЕ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ВОЛОНТЁРСКОЙ ОРГАНИЗАЦИИ

А. С. Трохин, Н. Р. Потапов

Воронежский государственный университет

Введение

С развитием производственных технологий большинство предметов первой необходимости становится всё доступнее. При этом остается значительная часть населения, финансовые возможности которых весьма ограничены. Возникает потребность, во-первых, обеспечить государство средствами поиска людей за чертой бедности для последующей помощи им. Во-вторых, необходимость в удобном интерфейсе для работы с данными, разработке дополнительных инструментов решения характерных для данной области проблем, а также инструментов коммуникации организаций, занимающихся подобной деятельностью. Программное приложение для волонтерской организации должно обладать понятным графическим интерфейсом, должно быть доступным для комфортного использования как можно большему количеству пользователей, функционировать на всех существующих системах, иными словами – кроссплатформенным. Рассматриваемый модуль приложения состоит из пяти компонентов.

Компонент «Новостная лента» должен обеспечить следующие возможности:

1. Возможность просмотра опубликованных запросов.
2. Возможность создания запросов от анонимных пользователей.
3. Возможность создания запросов от зарегистрированных пользователей.
4. Возможность фильтрации запросов по городу назначения.

Компонент «Пользовательские взаимодействия» должен обеспечить следующие возможности:

1. Возможность самостоятельной регистрации пользователя в приложении.
2. Автоматическая генерация уведомлений, отображаемых зарегистрированному пользователю на UI, о следующих событиях:
   a. Запрос одобрен.
   b. Запрос отклонен.
   c. Запрос нуждается в редактировании.
3. Возможность просмотра профиля авторизованного пользователя.

Компонент «Модерация запросов» должен обеспечить следующие возможности:

1. Возможность модерации запросов, созданными пользователями к компоненте «Новостная лента», следующими способами:
   a. Размещение запроса в новостной ленте.
   b. Отклонение запроса с просьбой редактирования своего запроса.
   c. Отклонение запроса без просьбы.
   d. Редактирование запроса модератором (для анонимных пользователей).
2. Отправка почтовых уведомлений зарегистрированным пользователям, о следующих событиях:

a. Отклонение запроса.
b. Публикация запроса.
c. Отклонение запроса с просьбой, редактирования.

Компонент «Администрирование» должен обеспечить следующие возможности:

1. Возможность просмотра информации о зарегистрированных в системе пользователях.
2. Возможность изменения/удаления информации о зарегистрированных в системе пользователях.

Компонент «Платежная система» должен обеспечить следующие возможности:

1. Возможность прикреплять к посту возможность оплаты (для участника волонтерской организации)
2. Возможность перевода денег через платежную систему
3. Возможность отслеживать процесс сбора средств по определенному запросу

1. Этапы разработки

Разработку web-приложений можно разделить на несколько этапов, часть которых может отсутствовать или разделена на более узкие разделы в зависимости от требований конкретного проекта.

1. Проектирование веб-приложения.
2. Разработка и создание дизайн-концепции сайта.
3. Front-end разработка:
   a. Создание макетов страниц.
   b. Создание мультимедиа и содержимого страниц.
   c. Вёрстка страниц и шаблонов.
4. Back-end разработка:
   a. Программирование или интеграция в систему управления содержимым (CMS).
   b. Обеспечение слаженной работы между front-end и back-end компонентами проекта.
5. Оптимизация и размещение материалов сайта.
6. Тестирование и внесение исправлений.
7. Публикация проекта и материалов на сервере.
8. Обслуживание и поддержка разработанного программного обеспечения.

2. Анализ front-end фреймворка Angular2

2.1. Краткое описание

Клиентская часть создана на основе фреймворка Angular, версия 7. Фреймворк Angular является компонентно-ориентированным фреймворком со строгой типизацией, которая является одним из главных достоинств фреймворка, позволяющим создавать строго типизированный код со старта разработки без дополнительных настроек. Другим несомненным преимуществом является механизм «инъекции зависимостей», позволяющий создавать слабосвязанный код.

2.2. Инструментарий

Удобным инструментом является использование командной строки Angular-cli. С его помощью можно сгенерировать как начальный шаблон проекта с готовой конфигурацией и иерархией файлов, так и сами компоненты фреймворка. Angular-cli поддерживает шаблоны под следующие виды angular-компонентов:

1. Class.
2. Component – главный строительный блок приложения.

3. Directive – компонент отвечающий за прямые манипуляции с DOM (хотя они и нежелательны).

4. Enum.

5. Guard – компонент отвечающий за безопасность, проверка определенных условий при навигации в приложении.

6. Interceptor – перехватчик запросов, производит обработку перед отправкой запроса

7. Pipe – компонент отвечающий за предварительную обработку данных в шаблоне компонента.

8. Service – сервисный компонент для хранения бизнес-логики отдельно от логики представления.

Поддерживаются и другие шаблоны, которые не использовались при написании рассматриваемого приложения.

3. Анализ back-end фреймворка Spring Boot

3.1. Краткое описание

Серверная часть написана на языке Java11, с использованием фреймворка Spring Boot. Были использованы следующие модули:

1. Spring Data JPA. Модуль для взаимодействия с базой данных, с помощью:

   a. JDBC напрямую, путём использования обёрток над стандартными JDBC подключениями, например JdbcTemplate.

   b. ORM фреймворка Hibernate, путём конфигурирования сущностью, которые описывают отображение столбцов базы данных в поля класса в Java.

2. Spring Security. Модуль безопасности, для обеспечения безопасного доступа к приложению с помощью авторизации пользователя в приложении. Также используется для шифрования информации пользователей, например паролей. В приложении использована простая система аутентификации – Basic Auth.

3. Spring MVC. Модуль для реализации паттерна Model-View-Controller. В данном приложении используются в основном rest-запросы, поэтому View является клиентская часть приложения.

4. Утилитные вещи, не выделенные в отдельный модуль. Например, пакеты для работы с сетью, были необходимы для реализации обращения к платежной системе.

3.2. Инструментарий

В качестве веб-сервера был использован встроенный в фреймворк сервер Apache Tomcat. Для хранения данных используется SQL база данных PostgreSQL. Несмотря на необходимость конфигурирования, она является предпочтительным вариантом, нежели более простые аналоги, например MySQL. В частности, СУБД PostgreSQL была выбрана из-за надежного механизма транзакций и возможности хранения слабоструктурированных данных, таких как JSON. Во время разработки приложения часто возникала необходимость работать с определенным набором тестовых данных. Для решения такой проблемы был использован инструмент Liquibase и его интеграция с фреймворком Spring. В нескольких SQL скриптах была описана схема базы данных и необходимые тестовые данные для приложения. При каждом запуске приложения схема базы очищалась и создавалась заново с помощью описанных ранее скриптов. Это существенно ускорило процесс тестирования и отладки приложения, в сравнении с тем, если бы те же самые данные приходилось вносить вручную.

В качестве механизма обмена данными, а также для отображения таблиц из базы данных в сущности был выбран наиболее удобный ORM framework hibernate. Его основными плюсами являются следующие свойства:

1. Бесплатный.
2. Обладает высокой производительностью.
3. Абстрагирует приложение от базовой базы данных SQL и SQL-диалекта.
4. Помогает сократить количество строк кода, делает систему более понятной и позволяет сосредоточиться на бизнес-логике, вместо написания SQL запросов.

4. Реализация проекта

4.1. Проблематика

После проведения сравнительного анализа с аналогичными приложениями данной предметной области были выявлены два основных недостатка. В первую очередь, необходимость регистрации в приложении. Не каждый пользователь готов предоставить кому-либо свои личные данные или даже потратить время на регистрацию фальшивой почты, чтобы пройти этап регистрации. Таким образом, если сделать регистрацию в системе необязательной, но возможной функцией, это сделает приложение более открытым для пользователя. Второй, и не менее важной проблемой является акцентирование внимания организаций на крупных проблемах и почти полное игнорирование, так называемых, точечных. Великие дела не делаются сразу[1], таким образом, дать возможность населению городов оказывать помощь своим землякам будет означать – решать локальные проблемы более эффективно. Две вышеописанные проблемы являются отправной точкой для работы над приложением.

4.2. Структуры данных приложения

Одной из основных сущностей и единицей информации в системе является пост или Запрос. Это простая структура, состоящая из следующих пользовательских полей:

1. Текст запроса, в котором пользователь описывает проблему.
2. Город публикации, для которого описанная проблема является актуальной.
3. Вложения при необходимости.

Также в структуре при сохранении заполняются следующие технические поля:

1. Уникальный идентификатор (в дальнейшем id) запроса.
2. id автора, если тот авторизованный пользователь системы, в противном случае это поле остается пустым, что является индикатором, того, что автор данного Запроса предпочел остаться анонимным.
3. id статуса запроса, при создании запрос находится в статусе «Новый».

Ещё одной важностью сущностью является «оплата», она крепится к постам и содержит упрощённую информацию о полученных средствах. Данная сущность состоит из следующих полей:

1. Описание, в котором рассказывается для чего создана данная «оплата»
2. Максимальная сумма или иначе говоря цель – количество денег которое необходимо набрать
3. Текущая сумма, описывающая количество денег набранных на данный момент
4. id поста к которому относится и в контексте которого рассматривается данная «оплата»

С помощью данной структуры пользователи приложения могут создавать Запросы, чтобы привлекать внимание людей к определенным проблемам в рамках конкретного города. Си-

[1]Цитата Софокла

стема статусов, о которой было сказано выше, необходима для более детального описания проблемы автором запроса. После попадания Запроса в систему, он отправляется на проверку волонтёрам организации, которые могут:

1. Отклонить пост, если он полностью не соответствует правилам организации.

2. Отправить пост на дополнительное редактирование, описав свои замечания в заметках, которые прикрепляются к посту, и будут видны пользователю во время последующего редактирования.

3. Согласовать публикацию поста, и при необходимости, прикрепить возможность денежного перевода, на проблему, описанную в запросе.

Каждое действие, связанное с переводом поста по статусам, фиксируется в базе данных. Таким образом, всегда можно будет узнать, историю развития запроса, или, например, получить информацию о том, какой волонтёр согласовал пост, не соответствующий правилам организации.

Заключение

Результатом исследования стал анализ средств front-end и back-end разработки веб-приложений. На основе анализа были выбраны инструменты необходимые для реализации проекта. Были разработаны и реализованы основные структуры данных в системе.

Литература

1. Документация Spring – Режим доступа: https://docs.spring.io/spring/docs/current/spring-framework-reference/index.html. (Дата обращения: 15.04.2020).

2. Документация Spring Boot – Режим доступа: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/ (Дата обращения: 15.04.2020).

3. Документация Angular2 – Режим доступа: https://angular.io/docs. (Дата обращения: 15.04.2020).

Трохин Антон Сергеевич – студент 4-го курса кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: toxatr0255@gmail.com

Потапов Никита Романович – студент 4-го курса кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: potapovnr@gmail.com

Горбенко Олег Данилович (научный руководитель) – канд. физ.-мат. наук, доцент, доцент кафедры математического обеспечения ЭВМ Воронежского государственного университета. E-mail: oleg_dan@mail.ru

УДК 004.9

СОЗДАНИЕ САЙТА КОНФЕРЕНЦИИ С ИСПОЛЬЗОВАНИЕМ CMS WORDPRESS

К. В. Турбина, М. А. Лаукарт

Воронежский государственный университет

Введение

Основная масса сайтов в недавнем прошлом была статичной и требовала внесения правок в их содержимое вручную. С текущей динамикой развития проектов необходима готовность быстро реагировать на изменения и внедрять их с максимальной оперативностью. При этом не все хотят или могут себе позволить обращаться к разработчикам, особенно если сайт требует постоянной работы над ним. Поэтому сейчас очень популярны технологии CMS. CMS – это система управления контентом, набор скриптов для создания, редактирования и управления контентом сайта. Такие системы позволяют пользователям, не имеющим навыков разработки самостоятельно вносить изменения в сайт. Примерами CMS являются WordPress, Tilda, OpenCart.

1. Постановка задачи

Рассмотрим текущую версию сайта конференции «Актуальные проблемы прикладной математики информатики и механики» [2] (рис. 1). Первая проблема данного сайта заключается в том, что данный сайт является статичным, но вносить изменения и вывешивать новости на нем необходимо регулярно, что, как было написано ранее, неудобно. Вторая проблема – это отсутствие личного кабинета. Это приводит к тому, что при внесении изменений в статью авторам необходимо связываться с организаторами конференции, а так же организаторам конференции необходимо вручную просматривать и обрабатывать множество заявок. Поэтому необходима база данных для сайта, с помощью которой будет проще обрабатывать необходимые данные. И третья проблема – это несовременный дизайн. Необходимо исправить эти проблемы.

Рис. 1. Главная страница старого сайта конференции

2. Решение

Устранение всех проблем возможно с помощью переноса сайта на Content management system (CMS, система управления контентом). В качестве CMS будем использовать WordPress (WP) [3].

2.1. Wordpress

Wordpress является самой популярной CMS. Статистика говорит, что на WP работает более 20 % всех сайтов в интернете, а также более 60 % ресурсов, построенных на CMS. Причина такой популярности в бесплатности, открытом коде и плагинах.

2.2. Создание сайта

Для исправления самой простой проблемы – несовременного внешнего вида, установим тему Museum Core. Добавим логотип факультета и фоновое изображение на заголовочную часть. В названии сайта укажем «Актуальные проблемы прикладной математики, информатики и механики». Создаем новые страницы: Архив конференции, Важные даты, Галерея, Гостиницы, Информационное письмо, Научные направления, Новости, О Конференции, Организационный комитет, Правила оплаты организационных взносов, Правила оформления статей, Программа конференции, Сборник трудов и другие материалы конференции.

Всю необходимую информацию перенесем со старого сайта.

Далее в настройках внешнего вида добавляем на шаблон меню со всеми страницами, группируя их. Также добавляем в шаблон контактную информацию организаторов конференции. Теперь главная страница выглядит так, как показано на рис. 2.

Рис. 2. Главная страница обновленного сайта конференции

2.3. Многоязычность

Для многоязычности нашего сайта установим плагин Polylang. Он даем возможность легко перевести страницы на английский язык и добавить виджет с выбором языка. После установки этого плагина в списке для каждой станицы появится датчик, переведена ли эта страница на нужные языки (рис. 3).

Рис. 3. Список страниц сайта

Переведем страницу главную страницу «Добро пожаловать», а также заголовок и виджеты на английский язык и получим вот такую главную страницу (рис. 4).

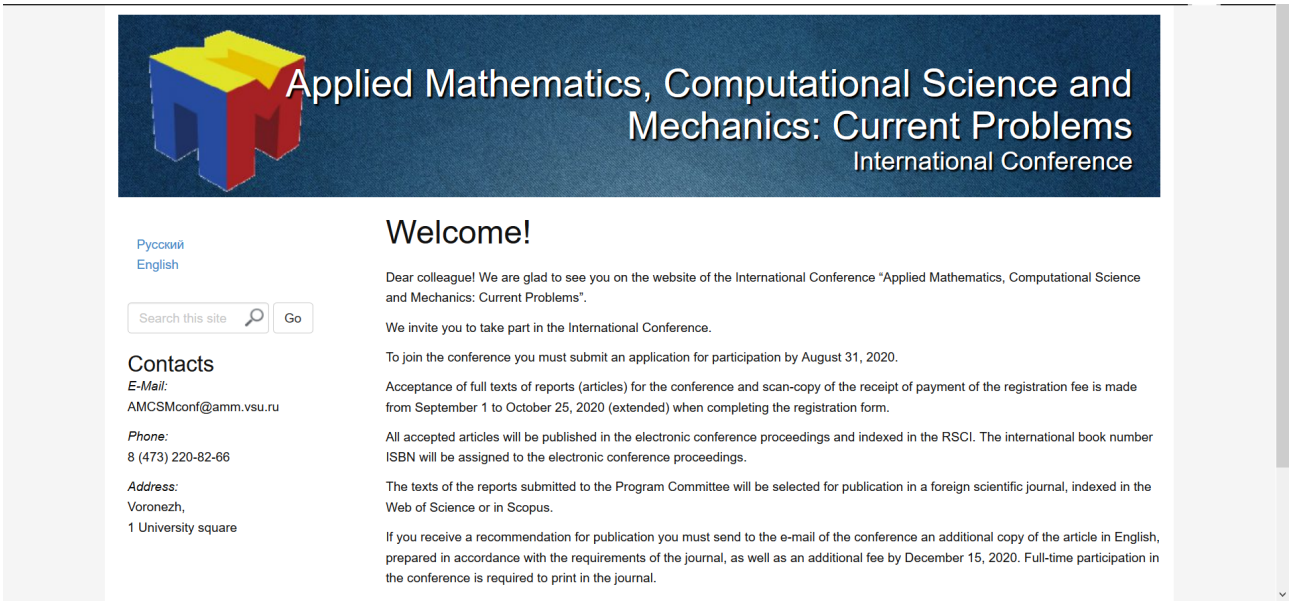


Рис. 4. Главная страница сайта на английском языке

Появился виджет переключения языков, а также появилась отметка о том, что страница переведена (рис. 5).

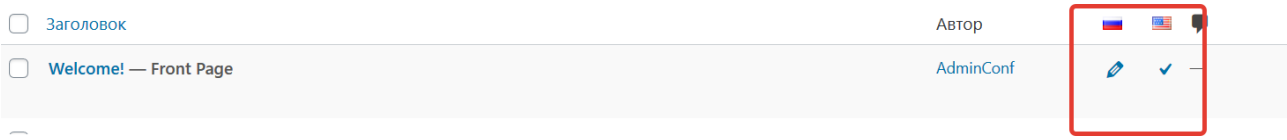


Рис. 5. Список страниц сайта

2.4. Галерея

Для создания галереи с фотографиями текущей и предыдущих конференций установим плагин Photo Gallery. Добавим фотографии в медиафайлы сайта. На странице «Галерея» добавим блок «Photo Gallery», выберем фотографии, которые хотим видеть в этой галерее, и нажмем опубликовать. Получим галерею (рис. 6).

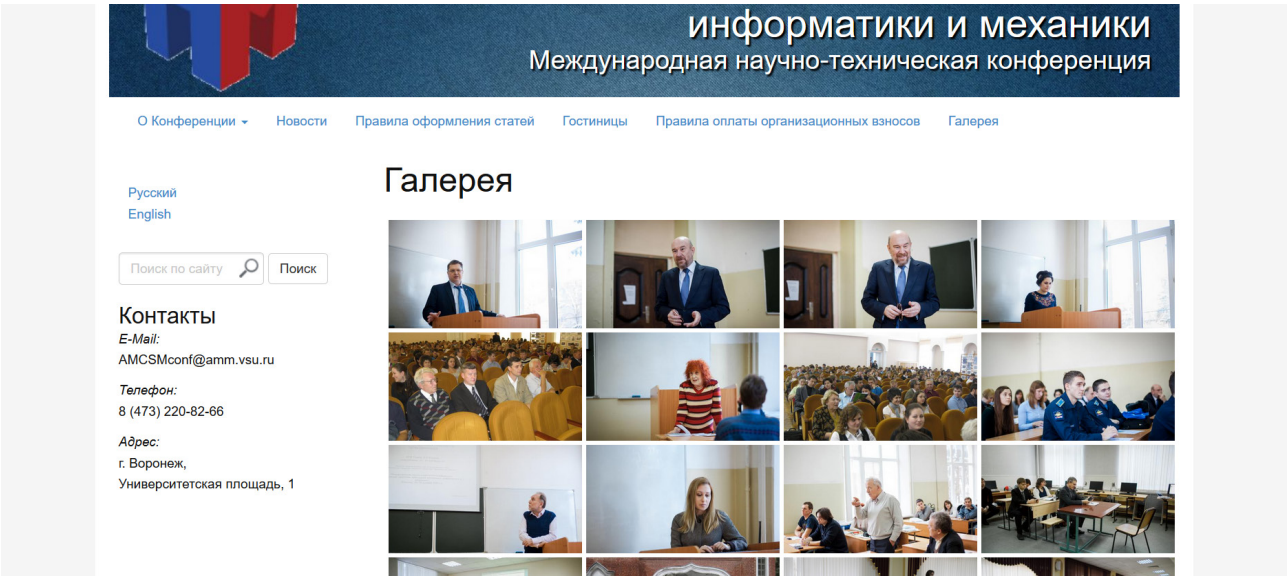


Рис. 6. Галерея

2.5. Личный кабинет

На текущем сайте нет личного кабинета, что доставляет трудности, как авторам, так и организаторам. Для внесения изменений в уже поданную статью авторам необходимо отправить письмо организаторам на почту. В связи с этим организаторам необходимо разбирать большой поток электронных писем, что очень нелегко. Также все данные об авторах хранятся в файлах Excel, которые обрабатываются вручную, что тоже занимает немало времени. Для решения этих проблем проведем следующие действия.

Сначала для формирования личного кабинета установим плагин WP-Recall. Однако использовать его будем только для входа, так как для обработки данных о докладах и авторах необходимы нестандартные (созданные вручную) таблицы базы данных, а с помощью этого плагина запись в такие таблицы совершить нельзя. Поэтому в базе данных создадим две таблицы: с авторами и с докладами. Дальше создаем форму для добавления статьи, где для каждой статьи указывается: тема статьи, выбор одной из списка секций, участие (очное или заочное), информация об авторах, а также прикрепляются файлы с докладом и подтверждение оплаты. Чтобы эти данные вносились в таблицы, созданные вручную, при нажатии кнопки «добавить статью» выполняется скрипт, который при помощи команды *wpdb* выполняет SQL-запросы внесения данных в базы.

Далее на странице «личный кабинет» создаем таблицу, в которой показаны доклады пользователя, авторы этого доклада и дата последних изменений. Теперь пользователь может менять данные о докладе без участия организаторов, а у организаторов к началу конференции будет вся необходимая информация об авторах без обработки большого объема данных.

Заключение

В данной работе отражены недостатки текущего сайта конференции «Актуальные проблемы прикладной математики информатики и механики» и показаны варианты их устранения на основе обновленного сайта. В нем были исправлены проблемы оперативного внесения новостей на сайт, добавлен личный кабинет пользователя, а также усовершенствован дизайн.

Литература

1. Сайт конференции «Актуальные проблемы прикладной математики информатики и механики». – http://www.amm.vsu.ru/conf/

2. Документация CMS WordPress. – https://codex.wordpress.org/ru:Main_Page

Турбина Ксения Владимировна – студент 3-го курса кафедры вычислительной математики и прикладных информационных технологий Воронежского Государственного Университета. E-mail: turbina2899@gmail.com

Лаукарт Мария Александровна – студент 3-го курса кафедры вычислительной математики и прикладных информационных технологий Воронежского Государственного Университета. E-mail: mariya.laukart@yandex.ru

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 378.18

ЮБИЛЕЙНЫЕ ДАТЫ РОССИЙСКОЙ ИНФОРМАТИКИ

Д. Г. Усков

Воронежский государственный университет

Предыдущий 2019 год был юбилейным для факультета прикладной математики, информатики и механики Воронежского государственного университета, которым был организован в нашей стране одним из первых факультетов подобного профиля, где информатика является одной из основных дисциплин.

Прошедший 2018 год был юбилейным не только для Воронежского государственного университета, который отметил в этом году свое столетие, но и для Российской информатики, днем рождения которой считается 4 декабря 1948 года [4, 6]. Государственный комитет Совета Министров СССР по внедрению передовой техники в народное хозяйство выдал 4 декабря 1948 года член-корреспонденту Академии наук СССР И. С. Бруку и молодому инженеру Б. И. Рамееву авторское свидетельство, подтверждающее изобретение цифровой вычислительной машины с приоритетом [4]. 15 декабря 1951 года зарегистрирована первая цифровая вычислительная машина М1, которая сконструирована в нашей стране студентами и выпускниками вузов.

Эти значимые события отмечены студенческими соревнованиями по информатике и программированию, которые регулярно проводятся на нашем факультете.

Межвузовская студенческая олимпиада по информатике, посвященная 30-летию факультета ПММ проведена 25 февраля 2000 года [1].

Олимпиада первокурсников вузов г. Воронежа, посвященная 70-летию одного из основателей факультета ПММ Д.Д. Ивлева, организована 20–22 сентября 2000 года.

Региональная студенческая интернет-олимпиада, посвященная 85-летию ВГУ состоялась в марте 2003 года.

VII региональная открытая школа-олимпиада по программированию и компьютерному моделированию, посвященная памяти первого декана факультета ПММ Г.И.Быковцева, проведена 16–17 сентября 2006 года согласно приказа Министерства образования и науки № 285 от 14.11.06 «О награждении победителей олимпиад» [1].

VI Всероссийская студенческая олимпиада «Информатика. Программирование. Информационные технологии», состоявшаяся 20–23 ноября 2008 года, была посвящена 60-летию Российской информатики и проведена согласно приказа Федерального агентства по образованию № 261 от 31.03.2008 в рамках реализации национального проекта «Образование» [3].

VII Всероссийская студенческая олимпиада «Информатика. Программирование. Информационные технологии», посвященная 40-летию факультета ПММ ВГУ, проведена согласно приказа Министерства образования и науки РФ № 254 от 13.03.2009 в октябре 2009 года.

Открытые on-line соревнования вузов Черноземья, посвященные 95-летию ВГУ, состоялись в декабре 2013 года.

Соревнования по программированию, посвященные 25-летнему юбилею ЗАО НПП «Релэкс» состоялись 9–16 декабря 2015 года [2].

Олимпиада студентов вузов г. Воронежа по информатике и программированию, посвященная 100-летию со дня рождения Селима Григорьевича Крейна, проведена 8 декабря 2017 года.

Соревнования первокурсников вузов г. Воронежа по информатике и программированию, посвященные 100-летию ВГУ состоялись 6 октября 2018 года.

Олимпиада по информатике и программированию студентов воронежских вузов, посвященная 70-летию Российской информатики, организована в апреле 2019 года.

В ноябре 2019 года успешно прошли соревнования студентов вузов г. Воронежа, посвященные полувековому юбилею факультета ПММ ВГУ.

В соревнованиях приняли участие 104 первокурсника, представляющих вузы:

- Воронежский государственный университет инженерных технологий (ВГУИТ),
- Воронежский государственный педагогический университет (ВГПУ),
- Воронежский институт высоких технологий (ВИВТ),
- Воронежский государственный университет (ВГУ),
- Военный учебно-научный центр Военно-воздушных сил «Военно-воздушная академия им. проф. Н. Е. Жуковского и Ю. А. Гагарина» (ВУНЦ ВВС ВВА)
- Воронежский государственный лесотехнический университет.

Победители олимпиады:

1 место

Плотников И. С. (ВГУ),
Плотникова И. Н. (ВГУ).

2 место

Плетнев О. Е. (ВУНЦ ВВС ВВА),
Кириллов И. С. (ВГУ).

3 место

Мельников Д. А. (ВГТУ),
Ефанькин В. В. (ВИВТ),
Попов В. Н. (ВГУИТ),
Шевцов Б. Е. (ВГПУ).

Отличительная особенность студенческих соревнований по информатике и программированию, которые организует факультет ПММ ВГУ, является то, что они стали платформой различных форм студенческого самоуправления, одной из которых является студенческий оргкомитет олимпиад. Сфера деятельности студенческого оргкомитета достаточно многообразна: мониторинг автоматической обработки электронной почты, встреча и регистрация участников соревнований, анализ решений, статистическая обработка результатов, индивидуальная работа с участниками соревнований, разбор олимпиадных заданий.

В числе первых разработчиков прикладной программы для решения дифференциальной краевой задачи второго порядка был Селим Григорьевич Крейн (1951 год) совместно с С. А. Авраменко и С. А. Богомолец. С. Г. Крейн, профессор, заслуженный деятель науки РФ, успешно работал в 60–80 годы в Воронежском государственном университете заведующим кафедрой уравнений в частных производных и воспитал несколько поколений математиков-профессионалов [5].

Литература

1. *Горбенко, О. Д.* Воронежский государственный университет – базовый вуз Всероссийской студенческой олимпиады «Информатика. Программирование. Информационные технологии» / О. Д. Горбенко, О. Ф. Ускова, А. И. Шашкин // Вестник Воронежского государственного университета. Сер. Проблемы высшего образования. – Воронеж, 2012. – № 2. – С. 51–56.

2. Столетию со дня рождения Селима Григорьевича Крейна посвящается / И. А. Бойченко, К. Е. Селезнев [и др.] // Информатика : проблемы, методология, технологии. Информатика в образовании : материалы 18-й Международной школы-конференции, Воронеж, 8–9 февраля 2018 г. : в 7 т. – Воронеж, 2018. – Т. 7.– С. 13–17.

3. Шестидесятилетию Российской информатики посвящается / О. Ф. Ускова, О. Д. Горбенко, Д. Р. Лапыгин [и др.] // Актуальные направления научных исследований XXI века: теория и практика : сборник научных трудов по материалам международной заочной научно-практической конференции. – Воронеж, 2014. – № 4, ч. 2 (9-2). – С. 57–59.

4. *Ускова, О. Ф.* Российской информатике 70 лет / О. Ф. Ускова, О. Д. Горбенко, Н. А. Каплиева // Актуальные проблемы прикладной математики, информатики и механики: сборник трудов Международной научной конференции, Воронеж, 17–18 декабря 2018 г. – Воронеж, 2019. – С. 1416–1418.

5. *Костин, В. А.* С. Г. Крейн и цепная математическая реакция в Воронеже. / В. А. Костин, Д. В. Костин. – Воронеж : Научная книга, 2018. – 34 с.

6. *Усков, Д. Г.* Становление отечественной информатики / Д. Г. Усков, Н. А. Каплиева, Е. И. Федорова // Актуальные проблемы прикладной математики, информатики и механики: сборник трудов Международной научной конференции, Воронеж, 11–13 ноября 2019 г. — Воронеж, 2020. – С. 1716–1717.

Усков Даниил Геннадьевич – студент 3-го курса кафедры МО ЭВМ Воронежского государственного университета. E-mail: uskov.dan@mail.ru

Трофименко Елена Владимировна (научный руководитель) – канд. физ.-мат. наук, доцент кафедры МО ЭВМ Воронежского государственного университета. E-mail: evtrof@gmail.com

УДК 004.921

ОПРЕДЕЛЕНИЕ СТОЛКНОВЕНИЯ ДВУХ ОБЪЕКТОВ

М. И. Фалалеев

Воронежский государственный университет

Введение

В наше время компьютеры представляют собой мощные вычислительные машины, способные выполнять миллионы операций в секунду. И естественно не обойтись без симуляции реального или игрового мира. Одна из задач компьютерного моделирования и симуляции состоит в определении столкновения двух объектов, одно из решений которой реализуется теоремой о разделяющей оси.

1. Теорема о разделяющей оси

1.1. Общая теория

Теорема. Две выпуклые геометрии не пересекаются, тогда и только тогда, когда между ними существует гиперплоскость, которая их разделяет. Ось ортогональная разделяющей гиперплоскости называется разделяющей осью, а проекции фигур на нее не пересекаются [1].

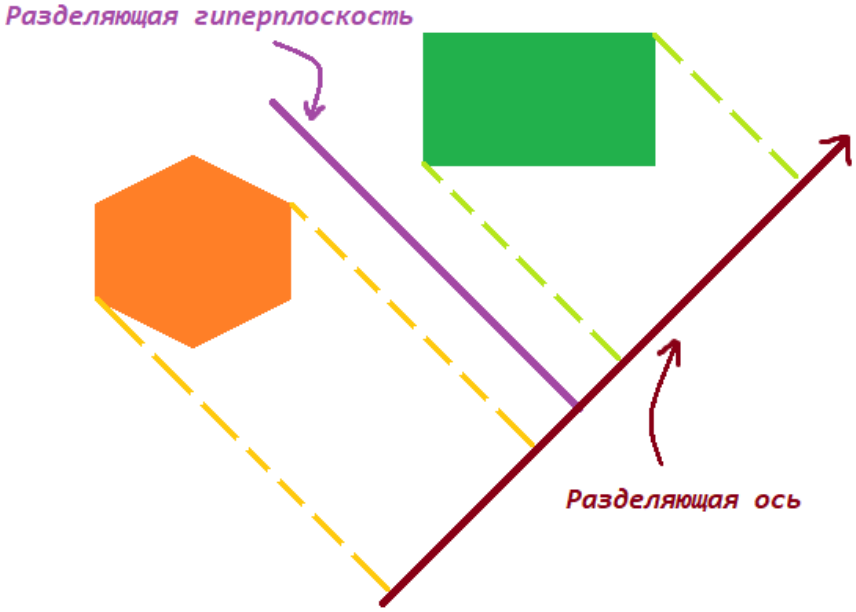


Рис. 1. Разделяющая ось (двумерный случай)

На (рис. 1, 2) видно, что проекции выпуклых объектов на разделяющую ось не пересекаются.

Свойство. Потенциальная разделяющая ось может находиться в следующих множествах (рис. 3):

1) нормали плоскостей каждого куба (красные);

2) векторные произведения ребер кубов $\{[\vec{x},\vec{y}] : x \in X,\ y \in Y\}$,

где X – ребра первого куба (зеленые), а Y – второго (синие).

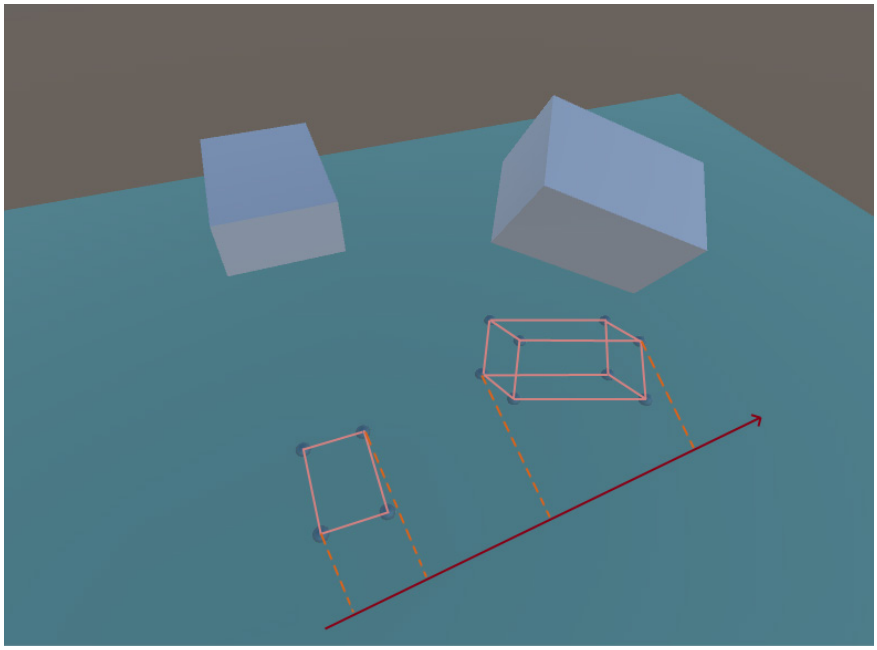


Рис. 2. Разделяющая ось (трехмерный случай)

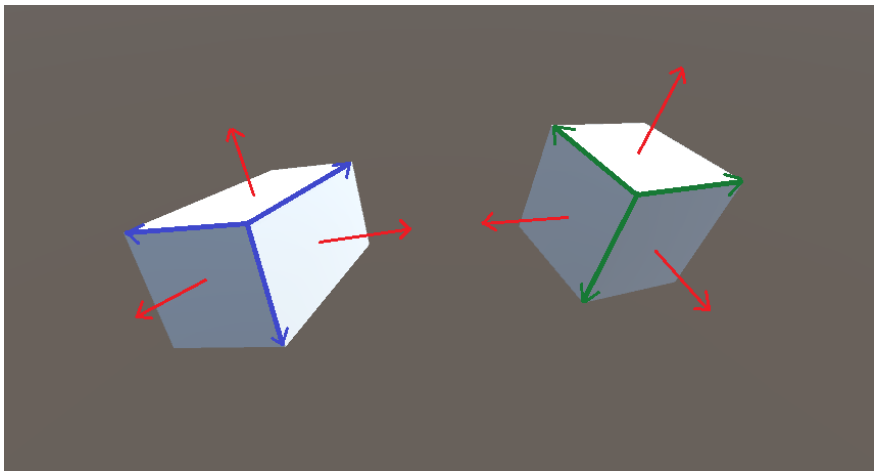


Рис. 3. Нормали и вектора для произведения

Следствие. Если объекты пересекаются, то из множества осей-кандидатов, та ось, которая дает минимальную длину пресечения проекций объектов, является направлением пересечения, а длинна отрезка – глубиной проникновения.

1.2. Алгоритм

Алгоритм проверки пересечения объектов представлен блок-схемой, изображенной на рис. 4.

2. Реализация

Примечание 1. В статье будет приведен пример с 2 параллелепипедами (далее – кубы), но идея, описанная ниже, будет сохранена и для других выпуклых объектов.

2.1. Кватернионы

Для начала необходимо определить функцию вращения вектора на кватернион [3].
Формула вращения вектора:

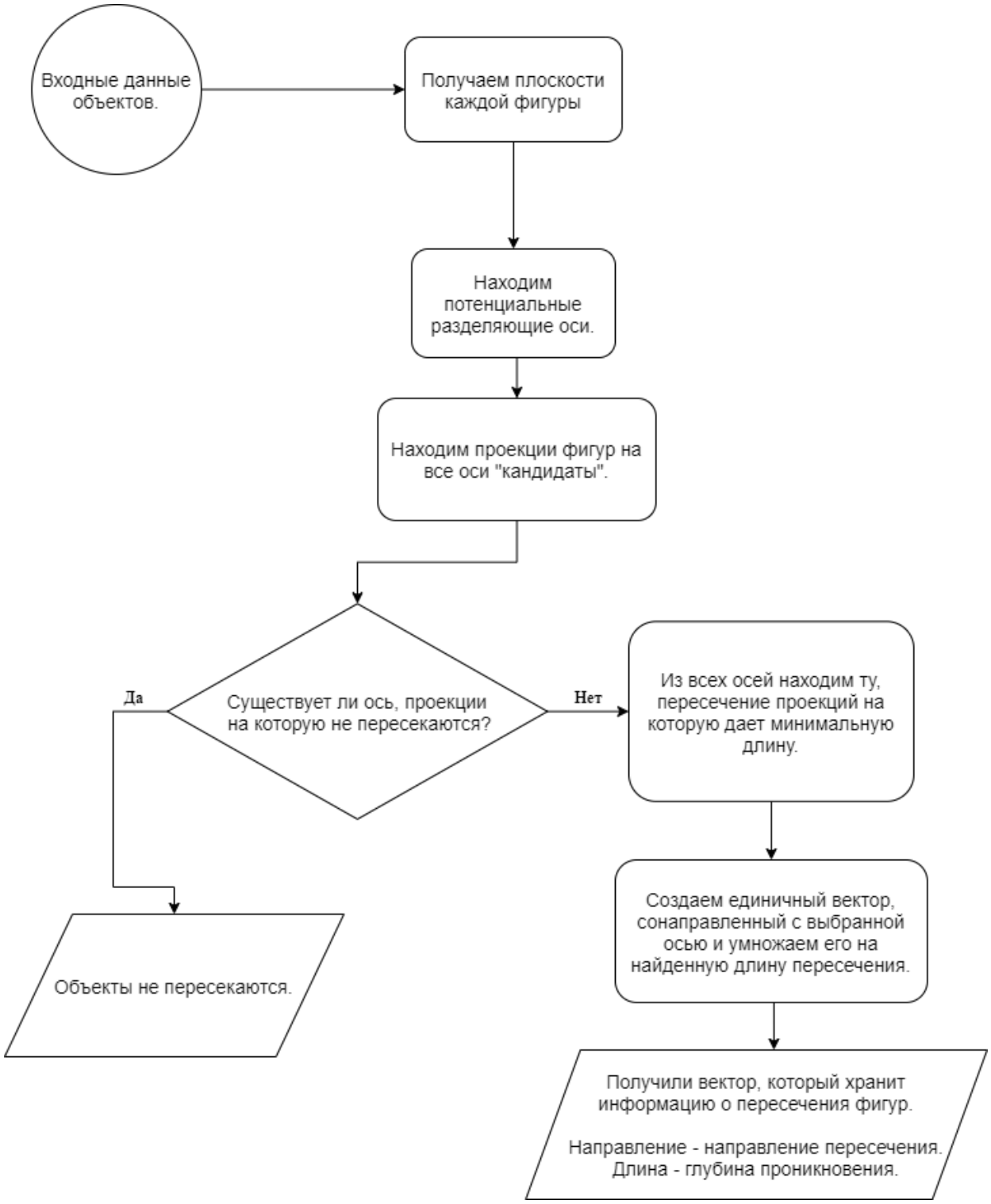


Рис. 4. Блок-схема алгоритма

$$\vec{v}' = q \times \vec{v} \times \overline{q},$$

где $\vec{v}'$ – новый вектор, $\vec{v}$ – исходный вектор, q – кватернион, $\overline{q}$ – обратный кватернион.

Распишем первую часть произведения

$$M = \vec{v} \times \overline{q} = \left(0 + v_x i + v_y j + v_z k\right) \times \left(q_w + q_x i + q_y j + q_z k\right) = v_x q_w i + v_x q_x - v_x q_y k + v_x q_z j +$$
$$+ v_y q_w j + v_y q_x k + v_y q_y - v_y q_z i + v_z q_w k - v_z q_x j + v_z q_y i + v_z q_z.$$

В результате получен кватернион

$$M = u_w + u_x i + u_y j + u_z k,$$

где $u_w = v_x q_x + v_y q_y + v_z q_z,$

$u_x i = v_x q_w i - v_y q_z i + v_z q_y i,$

$u_y j = v_x q_z j + v_y q_w j - v_z q_x j,$

$u_z k = -v_x q_y k + v_y q_x k + v_z q_w k.$

Вычислим оставшуюся часть

$$\overline{v}' = q \times M = \left(q_w + q_x i + q_y j + q_z k\right) \times \left(u_w + u_x i + u_y j + u_z k\right) =$$

$$= q_w u_x i + q_w u_y j + q_w u_z k + q_w u_w + q_x u_w i + q_x u_y k - q_x u_z j + q_x u_x +$$

$$+ q_y u_w j - q_y u_x k + q_y u_z i + q_y u_y + q_z u_w k + q_z u_x j - q_z u_y i + q_z u_z.$$

По определению кватернион – это вектор с нулевой скалярной частью, учитывая это, составим компоненты итогового вектора

$$\overline{v}' = v_x i + v_y j + v_z k$$

$$v_x i = q_w u_x i + q_x u_w i + q_y u_z i - q_z u_y i$$

$$v_y j = q_w u_y j - q_x u_z j + q_y u_w j + q_z u_x j$$

$$v_z k = q_w u_z k + q_x u_y k - q_y u_x k + q_z u_w k.$$

2.2. Получение вершин

Для того чтобы определить плоскости и ребра куба, нам понадобиться получить координаты его вершин.

Каждый куб может быть описан, например, следующими входными данными:

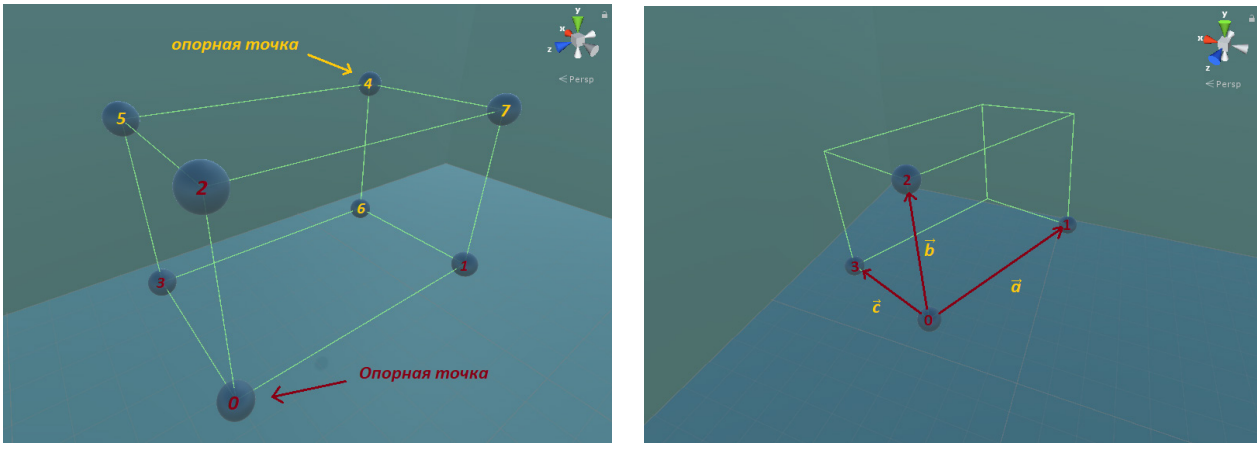
1) координаты центра;

2) размеры (высота, ширина и глубина);

3) кватернион вращения.

Для определения координат вершин необходимо найти две из них, которые формируют диагональ куба, назовем их опорными вершинами. Для получения первой точки к координатам центра прибавляем половину размерности куба, а для второй – вычитаем. Имея диагональные вершины, можно найти их соседние точки. Путем сложения или вычитания из каждой опорной вершины соответственно по каждой размерности куба.

На рис.5 а) и б) можно увидеть сформированные вершины и вектора-ребра.



а) *б)*

Рис. 5. а) вершины куба, б) основные вектора куба

Далее поворачиваем каждый вектор на соответствующий кватернион.

Далее, с помощью векторных произведений найдем потенциальные разделяющие оси.

2.3. Проекции на оси

На данном этапе необходимо найти проекции кубов на все потенциальные разделяющие оси. Ниже приведена формула, по которой можно узнать длину проекции вектора $\vec{v}$ на единичный вектор $\vec{a}$

$$proj_{\langle \vec{a} \rangle} \vec{v} = \frac{(\vec{v}, \vec{a})}{|\vec{a}|}.$$

Далее необходимо найти проекции кубов на каждую из потенциальных осей (рис. 6).

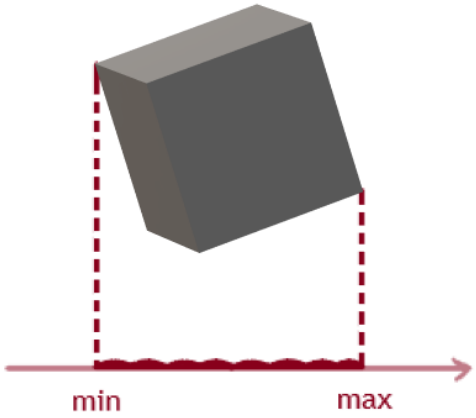


Рис. 6. Проекция куба на ось

Получив проекционные точки (рис. 7) каждого куба, необходимо определить пересекаются или нет проекции.

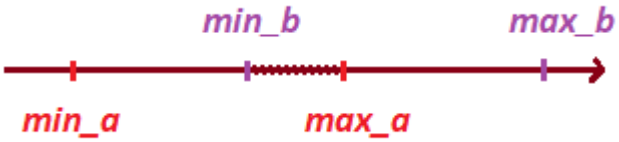
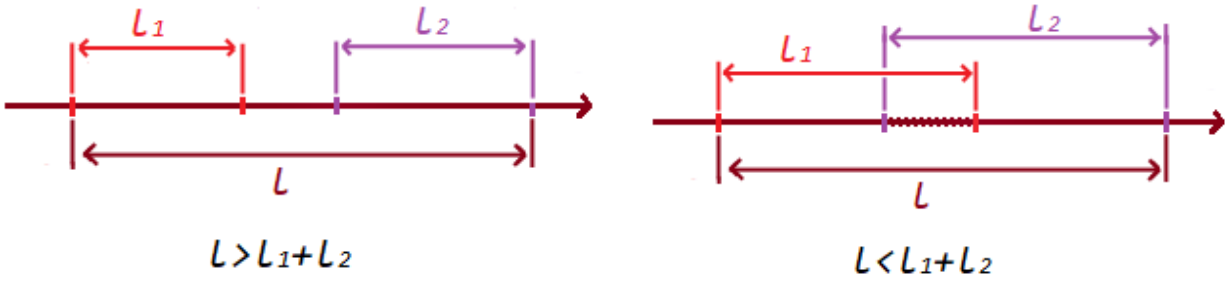


Рис. 7. Проекционные точки

Заметим следующие свойства.

1) Если отрезки не пересекаются, то сумма отрезков будет меньше, чем отрезок сформированными крайними точками (рис. 8 а)).

2) Если отрезки пересекаются, то сумма отрезков будет больше, чем отрезок сформированными крайними точками (рис. 8 б)).



а) *б)*

Рис. 8. а) случай не пересечения б) случай пересечения

Таким образом проверяем каждую ось на пересечение проекций.

Если будет найден хоть одна ось, проекции кубов на которую не пересекаются, то тогда и сами кубы не будут пересекаться. Поэтому, когда найдется разделяющая ось, можно уже не проверять оставшиеся вектора, и завершить работу алгоритма.

В случае пересечения кубов, все немного сложнее: проекции кубов на все вектора будут пересекаться, и необходимо будет определить вектор с минимальным пересечением. Это возможно сделать перебором. А именно, найти ось, проекции на которую пересекаются с минимальной длиной пересечения. Создать вектор, который будет сонаправлен с такой осью, и иметь длину равную длине пересечения проекций. Данный вектор будет полностью хранить информацию о пересечении.

Данный алгоритм был реализован в Unity [2]. Проект с реализацией и примером загружен на GitHub, и ознакомиться можно с ним по ссылке: https://github.com/slupok/BoxCollisionDetected

Заключение

Теорема о разделяющей оси может быть применена в определении столкновений двух тел в пространстве. Заметим, что данный алгоритм необходимо применять к геометрическим примитивам, описывающим сложные с большим количеством полигонов или сторон объектов.

Алгоритм не может быть применен если, хоть один из рассматриваемых объектов является не выпуклым (рис. 9). В этом случае для применения алгоритма потребуется разделить объект на множество выпуклых фигур.

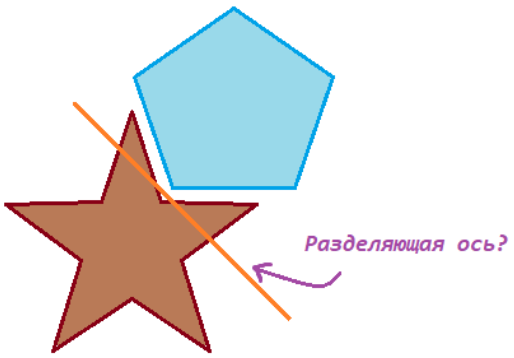


Рис. 11 Разделяющая ось для невыпуклых тел

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н.

Литература

1. Hyperplane separation theorem. – Режим доступа: https://en.m.wikipedia.org/wiki/Hyperplane_separation_theorem (Дата обращения: 05.04.2020).

2. Unity 3D. – Режим доступа: http://www.unity3d.ru/ (Дата обращения: 20.03.2020).

3. Кватернионы и бикватернионы с приложениями в геометрии и механике / В. Н. Гордеев. – Киев: Издательство «Сталь», 2016. – 316 с.

Фалалеев Максим Игоревич – студент 2-го курса направления прикладная математика и информатика факультета ПММ Воронежского государственного университета. E-mail: slupoke@yandex.ru

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 004.428.4

СОЗДАНИЕ БИБЛИОТЕКИ ДЛЯ АНАЛИТИКИ iOS ПРИЛОЖЕНИЯ

А. С. Филимонов

Воронежский государственный университет

Введение

С развитием современных технологий, рынок программных продуктов получает мощнейший стимул к расширению и развитию. Однако, сильнее всего растет рынок мобильных продуктов: смартфонов и планшетных компьютеров. И именно за этот рынок отвечает мобильная разработка, которая помогает бизнесу продвигать товары и услуги широкой аудитории. Но после того, как написан код, работа не заканчивается. Специалисты должны знать сильные и слабые стороны сервиса, маркетологи должны выстроить стратегию по распространению приложений конечным пользователям. Именно аналитика может дать ответы на все эти вопросы.

Актуальность данной задачи в том, что практически все разрабатываемые социальные приложения имеют в своем составе средства для сбора аналитики. Как правило, это либо общеизвестные, либо самописные библиотеки, которые очень негибкие и ограниченные. Это приводит нас к тому, что для того, чтобы внедрить сбор аналитики, необходимо пройти через несколько стадий, одна из которых – интеграция сбора аналитики в приложения.

Цели и задачи

Целью данной исследовательской работы является создание библиотеки для интеграции различных сервисов аналитики в iOS приложения.

Задачи данного исследования ставятся следующие:

- произвести анализ существующих библиотек аналитики;
- разработать гибкую библиотеку для интеграции сервисов аналитики.

Базовые понятия

В аналитике существуют базовые понятия.

Событие – это именованное сообщение, которое происходит в определенным момент времени и может содержать в себе полезную нагрузку в виде каких-либо key-value параметров.

Сервис аналитики – это сервис, который обеспечивает сбор и анализ результатов для дальнейшего анализа.

Существует большое количество сервисов, различающихся возможностями анализа и отображения результатов, но сбор, как правило, происходит везде одинаково – на основе событий.

Когда происходит какое-либо событие на клиенте, то на сервис аналитики должно быть послано сообщение с идентификатором и полезной нагрузкой события.

Анализ мобильных инструментов

Самые популярные мобильные платформы на рынке это iOS и Android. Но анализ мобильных инструментов будет проводиться именно на платформе от компании Apple – iOS, но на платформе Android ситуация схожая.

Обычно используют несколько сервисов для аналитики, поэтому необходимо абстрагироваться от какого-либо одного и иметь возможность одновременно отправлять события и полезную нагрузку в несколько различных сервисов.

Необходимо выявить требования, предъявляемые к библиотеке:

- максимальная независимость от проекта – в библиотеке должно содержаться максимально возможное количество проектонезависимого кода;
- простота интеграции – интеграция в проект должна быть быстрая и не должна занимать много кода;
- гибкость – должна быть возможность отправлять события не на все сервисы, а только на некоторые, а так же названия событий между различными сервисами могут не совпадать;
- простота использования – на каждое событие должна быть возможность отправлять полезную нагрузку;
- группировка событий – в проекте может быть несколько сотен событий и без группировки события будет сложно поддерживать и использовать.

Посмотрим и проанализируем существующие open-source библиотеки на соответствие предъявляемых требований с помощью табл. 1:

- Analytical;
- AnalyticsKit;
- Umbrella.

Таблица 1

Сравнение существующих библиотек для аналитики

	Analytical	AnalyticsKit	Umbrella
Независимость	+	+	+
Простота интеграции	+	+	+
Гибкость	–	+/–	–
Простота использования	–	–	+/–
Группировка событий	+	–	–

После анализа таблицы становится очевидно, что ни одна из рассмотренных библиотек не удовлетворяет всем требованиям.

Основные элементы библиотеки

Для удовлетворения всем требованиям библиотеки были выделенные следующие сущности, изображенные на рис. 1:

- AnalyticsManager – синглтон объект, которые является главным объектом в аналитике. Именно он отвечает за взаимодействие событий и сервисов.
- AnalyticsProvider – это протокол, который отображает сам сервис аналитики. Каждый сервис должен реализовывать протокол для связи с событиями.
- AnalyticsEvent – это протокол, который отвечает за событие. Данный подход позволяет реализовывать события как в виде перечисления, так и в виде класса и структуры. Именно этот факт обеспечивает ту гибкость, которая требовалась.

Проанализируем все требования и с помощью каких компонентов они реализуются в данной библиотеке:

- Максимальная независимость от проекта – все протоколы и AnalyticsManager вынесены в библиотеку, а значит этот код не придется переносить из проекта в проект. Вся библиотека поделена на 3 части: проектно-независимая часть, проектно-переиспользуемая часть и

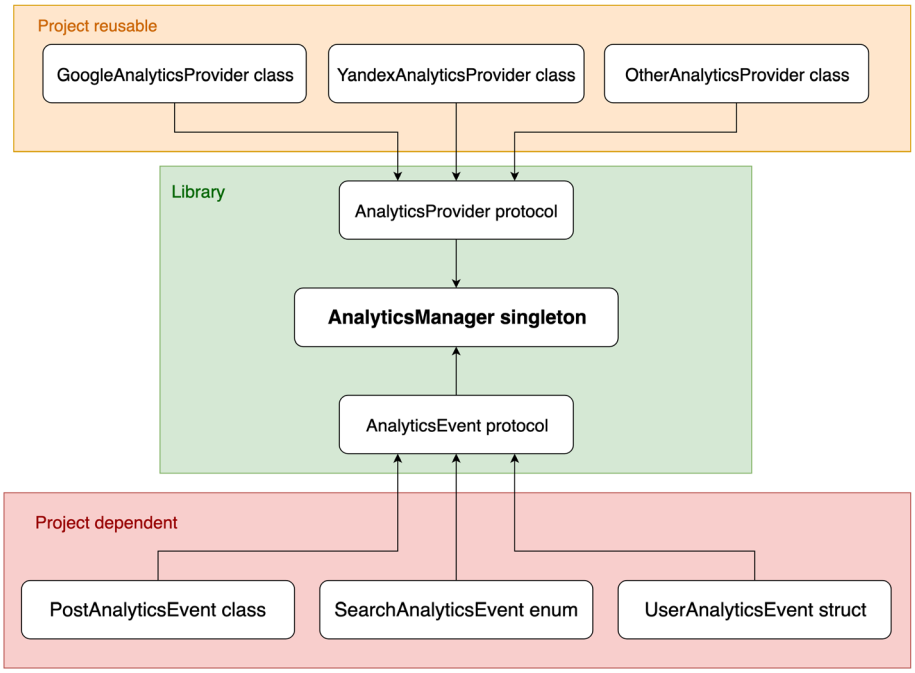


Рис. 1. Основные компоненты библиотеки аналитики

проектно-зависимая часть. AnalyticsProvider является проектно-переиспользуемой частью, что дает возможность переносить уже ранее реализованные сервисы между проектами. AnalyticsManager и протоколы являются проектно-независимой частью. Сами события являются проектно-зависимой частью.

- Простота интеграции – в данном подходе единственное, что надо будет делать это писать сами события аналитики, потому что сервисы будут перенесены из другого проекта, а менеджер будет подключен из библиотеки.

- Гибкость – в реализации AnalyticsEvent можно сделать так, чтобы различные события сами решали в какие сервисы им отправляться, а в какие нет, а так же можно назначит различным сервисам свои ключи как для названия событий, так и для полезной нагрузки.

- Простота использования – в реализации AnalyticsEvent можно будет сделать так, что полезная нагрузка будет передаваться в событие как параметры функций, а следовательно полезная нагрузка будет иметь тип.

- Группировка событий – реализация AnalyticsEvent позволяет делать группировку любого уровня вложенности.

Заключение

В результате исследования были проанализированы современные подходы к аналитике современных мобильных приложений.

В ходе исследования достигнуты следующие результаты:

- составлены требования, предъявляемые к библиотеке аналитики
- проведен анализ существующих библиотек для интеграции аналитики в приложения
- написана библиотека, которая полностью удовлетворяет всем требованиям.

Литература

1. Tracking Analytics. – Режим доступа: https://blog.basedata.solutions/tracking-analytics-with-swift-3225be77b976 (Дата обращения: 20.04.2020).

2. A modular analytics layer. – Режим доступа: https://www.lordcodes.com/articles/a-modular-analytics-layer-in-swift (Дата обращения: 15.04.2020).

3. Architecting an Analytics layer. – Режим доступа: https://medium.com/ios-os-x-development/architecting-an-analytics-layer-7cdacb5f74af. (Дата обращения: 21.03.2020).

4. Analytics Architecture Overview. – Режим доступа: https://www.essentialdeveloper.com/articles/clean-ios-architecture-part-1-analytics-architecture-overview?rq=analyti. (Дата обращения: 20.02.2020).

Филимонов Александр Сергеевич – студент 4-го курса кафедры МО ЭВМ Воронежского государственного университета. E-mail: as_filimon@mail.ru

Болотова Светлана Юрьевна (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры МО ЭВМ Воронежского государственного университета. E-mail: Bolotova.svetlana@gmail.com

УДК 004.023

ПОСТРОЕНИЕ ТРАЕКТОРИЙ ДВИЖЕНИЯ В РОБОТОТЕХНИКЕ ПОСРЕДСТВОМ МАШИННОГО ОБУЧЕНИЯ В СРАВНЕНИИ С ИНЫМИ ПОДХОДАМИ

К. С. Харина

Воронежский государственный университет

Введение

Развитие робототехники в последнее время становится особенно актуальным как никогда прежде. Ведь никогда прежде в новейшей истории при текущем уровне развития технологий еще не была так высока потребность в роботизации промышленности, сферы услуг и развлечений, сферы медицинского обслуживания, прочих социальных служб. Многие эксперты утверждают, что именно курс на роботизацию и автоматизацию сделают вышеперечисленные сферы устойчивыми к потрясениям. Но для успешного применения робототехники еще предстоит решить множество проблем.

Большую часть проблем в робототехнике можно сформулировать как проблемы, связанные с автономностью робота. Автономность робота в том числе подразумевает способность самостоятельно передвигаться. При текущем уровне развития науки и техники возможно как заранее программировать траекторию движения, так и позволять роботу обучаться и строить траектории самостоятельно. Но тут перед исследователем встает вопрос: какой подход, какой алгоритм использовать, чтобы максимально эффективно реализовать построение траекторий для робота? Этому вопросу посвящена данная статья.

1. Постановка задачи

1.1. Построение траекторий в робототехнике

В силу того, что любая автономная система, к коим относятся и различные роботы, для выполнения своих функций должна перемещаться в пространстве, возникает проблема построения траекторий движения.

Возьмем робоманипулятор из трех составных частей. Положим, что инженерные ограничения позволяют ему выстраивать свои траектории только относительно двух осей координат – x и y. Для выполнения любых манипуляций, для произведения которых существует данный класс роботов (манипуляторы), требуется построить кратчайшую траекторию, в которой заранее известна начальная точка A_1, которая совпадает с начальной точкой первого сегмента робоманипулятора (также – точка крепления), расположение в пространстве каждого из трех сегментов, обозначаемых как A_1A_2, A_2A_3, A_3A_4, и целевая точка B.

Требуется совместить точку A_4 и точку B путем построения наиболее оптимальной траектории движения робоманипулятора.

Критерии оптимальности:

1. Минимальное изменение положения каждого сегмента относительно начального – угол φ, на который будет изменять положение каждый из сегментов, должен быть минимальным.
2. Минимальное время на реализацию построения траектории.
3. Для методов машинного обучения – минимальное время на обучение.

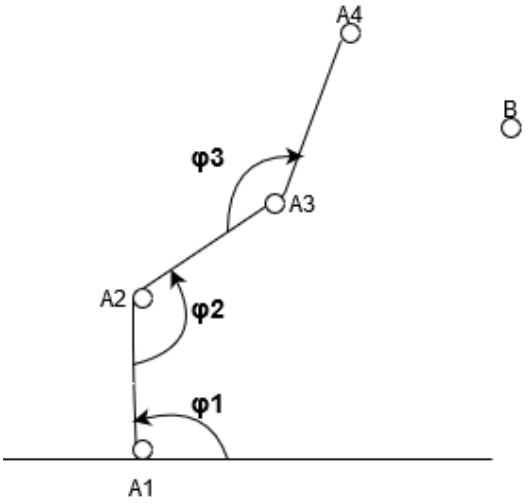


Рис. 1

1.2. Проблемы

Чтобы робот мог успешно работать в заданной среде, ему необходима построенная траектория для движения. Траектория может быть задана программным образом либо получена в результате применения машинного обучения.

Две ключевые проблемы в робототехнике:

1. Если траектория задана программным образом, то снижается гибкость применения робота в разнообразных средах, повышаются требования к окружению, в котором действует робот. Также требуется учитывать время, необходимое специалисту на написание программы траектории.

2. Если траектория получена в результате применения алгоритмов машинного обучения, то проблема заключается во времени на обучение робота и на принятие им решения. Требуются, с одной стороны, максимально быстродействующие алгоритмы, с другой стороны – предоставляемые ими результаты должны быть максимально близкими к оптимальным. До сих пор не найдено идеальное решение, в полной мере удовлетворяющее обоим этим критериям.

1.3. Недочеты распространенных подходов

Наиболее распространенный подход задания траектории для робота – задать ее программно. Недостатки данного подхода:

1. Длительный и дорогостоящий процесс: разработка алгоритмов планирования и управления движением для производственных роботов является длительным и утомительным процессом, требующим совместных усилий многих экспертов в данной области.

2. Отсутствие гибкости: заданная программно траектория обладает минимальными адаптационными свойствами и мало подходит для применения в сложных средах реального мира. Роботы, чьи траектории заданы программно написанными вручную алгоритмами, находятся под угрозой негативного воздействия любого из неучтенных факторов внешней среды – а такие факторы всегда имеют место быть, поскольку окружающая среда реально мира содержит, без преувеличения, бесконечное число переменных.

2. Методы решения

2.1. Глубокое обучение (обучение с подкреплением)

Исходя из того, что одна из проблем – наличие бесконечного числа переменных, составляющих окружающую среду реального мира, следует, что при подходе, когда робот строит свою траекторию самостоятельно, необходимо иметь возможность встраивать любую из этих переменных в применяемый для построения траектории алгоритм. Иными словами, робот должен уметь взаимодействовать с окружающей средой, получать от нее входные данные об окружающей среде, на их основе строить алгоритм и максимально быстро адаптировать его при получении новых данных или изменения состава переменных.

Именно такие задачи решает обучение с подкреплением (ОП) [1].

Однако при попытке применить ОП к реальному миру в контексте поставленной задачи возникают следующие проблемы:

1. Обучение занимает слишком много времени. Согласно критериям оптимальности процесса построения траектории, время обучения должно быть меньше или равно времени реакции робота на изменение обстановки, появления нового фактора (переменной) или исчезновения старого фактора. В настоящее время, на текущем уровне технического развития вычислительных мощностей, этот баланс труднодостижим.

2. Обучение с подкреплением подразумевает элемент эксперимента во взаимодействии робота с окружающей средой. Однако, если взаимодействие будет неудачным, то робот может быть поврежден или даже уничтожен (например, если при попытке построить траекторию робот получит новую входящую переменную в виде возникшего препятствия, на обучение будет затраченно времени больше, чем требуется для реагирования, и, двигаясь по старой траектории, робот повредится о препятствие). В качестве альтернативы, можно пытаться использовать симуляцию для обучения агента, а затем развертывать обученную политику на реальном роботе. Но имитировать хитросплетения реального мира очень сложно, и такая политика часто плохо работает на физическом роботе.

2.2. Генетические алгоритмы и эволюционные стратегии

В эволюционных стратегиях (ЭС) [2] сначала создается множество случайно сформированных объектов с заданной структурой (вариантов траектории, в данном случае), – такое множество называется популяцией. Далее все эволюционные стратегии работают по общей схеме: определяется пригодность объектов в популяции и функция, определяющая близость объекта к истинному решению, называемая целевой функцией. Далее определяется пригодность объектов в популяции, с учетом близости к целевой функции и при внесении элемента случайности создаются объекты для популяции, с учетом близости к целевой функции и при внесении элемента случайности создаются объекты для популяции следующей итерации. Данный процесс повторяется до получения решения, либо до окончания времени, отведенного на решения.

Преимущества эволюционных стратегий:

- простота реализации;
- не требуется обратного распространения;
- легко масштабируется в распределенной среде вычислений;
- малое число гиперпараметров.

Касательно задачи построения траектории для робоманипулятора ЭС обладают неоспоримыми преимуществами в быстродействии по сравнению с ОП с силу специфики реализации.

3. Сравнение

Таблица 1

Сравнение обучения с подкреплением против эволюционных стратегий

Критерий	Эволюционные стратегии	Обучение с подкреплением
Требуется ли обратное распространение	В ЭС есть только прямое распространение алгоритма построения траектории, что делает код проще и выполняется быстрее. В случае ограниченной памяти также нет необходимости хранить эпизоды для будущего обновления.	При ОП требуется обратное распространение алгоритма построения траектории, особенно при использовании рекуррентных нейронных сетей. Результаты получаются точнее, но код исполняется дольше, обучение занимает больше времени.
Степень параллелелизма	В ЭС требуется лишь незначительный обмен данных между процессами – скалярные данные о вознаграждении. Поэтому в ЭС наблюдается линейный рост производительности по мере добавления вычислительных ядер.	В ОП требуется синхронизация полных векторов признаков (миллионы чисел).
Эффективность выборки	Есть возможность применять алгоритмы с любым количество испытаний, поскольку испытания проводятся постепенно и с каждым следующим шагом точность построения оптимальной траектории движения с высокой вероятностью повышается.	Если необходимо обучать политику с использованием физических роботов, нужно разработать алгоритмы, которые требуют небольшого количества испытаний, чтобы обучение было осуществимо.

Заключение

Данная работа предполагает, что подходы нейроэволюции могут быть конкурентоспособными с методами обучения с подкреплением на современных эталонных тестах среды агента, предлагая при этом существенные преимущества, связанные со сложностью кода и простотой масштабирования для крупномасштабных распределенных настроек. Также ожидается, что более результативную работу можно проделать, пересмотрев другие идеи из этой области работы, такие как методы косвенного кодирования, или изменив структуру сети в дополнение к параметрам.

Работа выполнена под руководством канд. физ.-мат. наук Медведева С. Н. .

Литература

1. *Гудфеллоу, Я.* Глубокое обучение / Я. Гудфеллоу, И. Бенджио, А. Курвилль – Москва: ДМК, 2018. – 651 с.

2. *Рутковская, Д.* Нейронные сети, генетические алгоритмы и нечёткие системы / Рутковская Д., Пилиньский М., Рутковский Л. — 2-е изд.. — М.: Горячая линия-Телеком, 2008. — 452 с.

Харина Ксения Сергеевна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: ksenya.k.amber@gmail.com

Медведев Сергей Николаевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: s_n_medvedev@mail.ru

УДК 004.4.277.4

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ СОВМЕСТНОГО СОЗДАНИЯ МУЗЫКИ

Е. А. Харченко, А. О. Шабанов

Воронежский государственный университет

Введение

При написании музыки на заказ и при работе в команде зачастую возникают проблемы, связанные со сложностью коммуникации музыкантов, обмена музыкальными дорожками и их прослушиванием в онлайн режиме.

Так, во время написания музыкальной композиции для заказчика должны быть учтены все его пожелания. Если клиенту не нравится звучание одной звуковой дорожки в треке, композитор должен каждый раз при внесении правок экспортировать весь проект (процесс рендеринга занимает от 2 до 15 минут в зависимости от сложности проекта), а затем выкладывать его в облачный сервис или прикреплять к сообщению в каком-либо чате. И зачастую для достижения компромисса требуется многократная пересылка, доставляющая большие неудобства.

Работа в команде также имеет свои особенности. В ходе написания музыки от всех участников команды могут поступать предложения по усовершенствованию композиции, а при возникновении вопросов к звучанию какого-либо инструмента в конкретном моменте воспроизведения часто требуется указать на проблемный участок со ссылкой на временную отметку этого момента.

Все имеющиеся программные продукты, которые можно адаптировать под решение обозначенных выше проблем, имеют ряд недостатков, включая отсутствие некоторых важных функций и удобного пользовательского интерфейса.

В результате возникла идея написания веб-сервиса, позволяющего реализовать функциональность, решающую проблемы коммуникации музыкантов при совместном написании музыки.

1. Анализ функциональности

Перед началом разработки был проведен анализ необходимой функциональности, в результате которого были выделены три возможные роли участника проекта: администратор, коллаборатор, гость. На основе анализа был создан набор Use Case диаграмм.

Гость может прослушивать аудиозаписи, комментировать и писать сообщения в чате проекта (рис. 1).

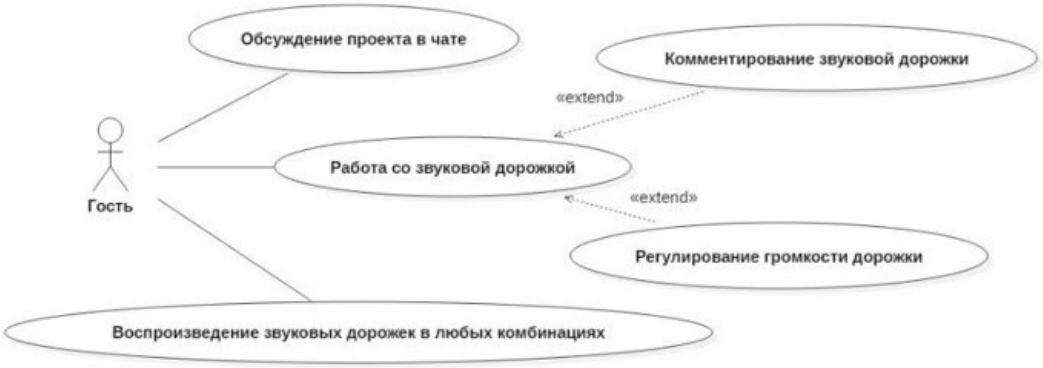


Рис. 1. Диаграмма вариантов использования для гостя

Коллаборатор может прослушивать аудиозаписи, комментировать и писать сообщения в чате проекта, добавлять новые дорожки, приглашать пользователей в проект (рис. 2).

Рис. 2. Диаграмма вариантов использования для коллаборатора

Администратор может управлять всем содержимым проекта и ролями приглашенных пользователей. Так же он имеет право удалить весь проект (рис. 3).

Рис. 3. Диаграмма вариантов использования для администратора

2. Реализация

Было реализовано web-приложение с использованием языков программирования Python, JavaScript и фреймворка Django. В качестве локального хранилища данных была использована PostgreSQL – свободная объектно-реляционная СУБД, которая хорошо масштабируется и поддерживает необходимые типы данных. Основными объектами, хранимыми в базе данных являются сущности «Пользователь», «Проект», «Звуковая дорожка».

Для работы со звуковыми дорожками была выбрана библиотека Pizzicato.js, построенная на Web Audio API. Она позволяет загружать файлы, менять громкости у дорожек и воспроизводить несколько дорожек одновременно.

При загрузке страницы для каждой звуковой дорожки создается объект. Для этого в конструктор передается ссылка на аудио файл. Полученный объект имеет параметр, отвечающий за громкость воспроизведения. При движении слайдера рядом с каждой звуковой дорожкой, изменяется значение соответствующего параметра.

Звуковые дорожки объединяются в группы. Каждая такая группа имеет методы для запуска, паузы и остановки воспроизведения звуков. Возможность включения и выключения звуковой дорожки при проигрывании реализуется путем установки громкости соответствующего объекта в минимум. При этом запоминается предыдущее значение громкости.

Так как библиотека не позволяет во время воспроизведения изменять временную метку, для реализации перемотки использована следующая логика: воспроизведение останавливается и начинается заново с определенного времени.

3. Интерфейс

Для приложения был разработан web-интерфейс, предоставляющий возможность использовать все заложенные функции. В качестве примера на рис. 4 приведен внешний вид страницы проекта для пользователя с ролью Администратор.

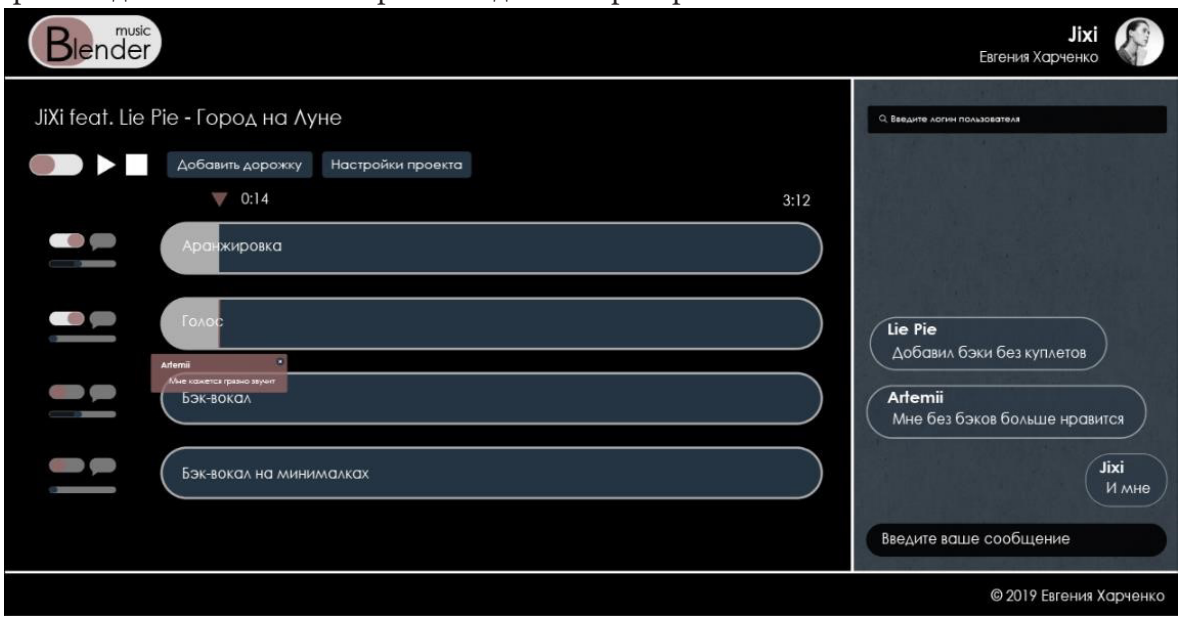


Рис. 4. Внешний вид страницы проекта для администратора

Заключение

В результате проделанной работы было создано веб-приложение, обладающее следующей функциональностью:

- регистрация, авторизация и аутентификация;
- создание проектов;
- предоставление к ним доступа другим пользователям;
- назначение пользователям ролей при работе с проектом;
- добавление и удаление звуковых дорожек;
- прослушивание одной, нескольких, или всех дорожек одновременно;
- комментирование дорожек;
- обсуждение процесса работы над композицией посредством проектного чата.

Полученное веб-приложение рассчитано на композиторов и звукорежиссеров и позволяет облегчить процесс коммуникации музыкантов при совместном написании музыки.

Литература

1. *Дронов, В. А.* Django: практика создания Web-сайтов на Python / В. А. Дронов. – Санкт-Петербург : БХВ-Петербург, 2016. – 672 с.

2. Django documentation. – Режим доступа: https://docs.djangoproject.com/en/2.2 (Дата обращения 14.02.2020).

3. Web Audio API. – Режим доступа: https://developer.mozilla.org/enUS/docs/Web/API/Web_Audio_API (Дата обращения 02.04.2020).

Харченко Евгения Андреевна – магистрант 1-го года обучения кафедры Программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: evgesha97@bk.com

Шабанов Артемий Олегович – магистрант 1-го года обучения кафедры Программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: artemshabanov7@mail.ru

Барановский Евгений Сергеевич (научный руководитель) – канд. физ.-мат. наук, доц., доцент кафедры Программного обеспечения и администрирования информационных систем Воронежского государственного университета. E-mail: esbaranovskii@gmail.com

УДК 658.7

РАЦИОНАЛИЗАЦИЯ ДВИЖЕНИЯ МАТЕРИАЛЬНЫХ ПОТОКОВ С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТОВ SAP MATERIALS MANAGEMENT

С. А. Хоф

Воронежский государственный университет

Введение

Планирование ресурсов предприятия (ERP) – это программное обеспечение, предназначенное для организаций, принадлежащих к различным отраслям промышленности, независимо от их размера и мощности.

ERP-системы предназначены для поддержки и интеграции практически всех функциональных областей бизнес-процессов, таких как закупка товаров и услуг, продажа и распределение, финансы, учет, управление персоналом, производство, планирование производства, логистика и управление складом [2, 3].

Системы управления, разработанные компанией SAP, предлагают SAP ERP, который охватывает различные процессы под одним и тем же интерфейсом. Эти процессы упрощены в их выполнении, но они могут обрабатывать сложные операции, предлагая большую специализацию с меньшими трудностями для пользователя.

Управление материальными потоками является одним из важных модулей в программном обеспечении SAP ERP. Модуль приложения MM поддерживает функцию инвентаризации закупок, выполняемую изо дня в день. Он в основном содержит заказ на поставку, получение товара, хранение материала и т. д.

Модуль SAP MM ускоряет процесс закупок благодаря работе с различными аспектами процесса, такими как основные данные материала и поставщика, определение счета и оценка материала, управление запасами, проверка счета-фактуры, планирование потребности в материалах и т. д.

1. Описание исследованной области

1.1. Управление материальными потоками

Одна из самых важных задач в управдении организацией является управление материальными потоками. При выполнении различных хозяйственных операций нужно учесть большое количество факторов, используя временные и финансовые ресурсы. Достаточно эффективно реализовать механизм синхронизации материальных потоков в логистической модели предприятия возможно при помощи модуля «Управление материальными потоками» (Materials Management) [4–6].

Использование SAP Materials Management (SAP MM) поддерживает эффективное управление хозяйственной деятельностью компании, благодаря следующим возможностям:

- учет объемов и стоимости складских запасов;
- контроль системы материально-технического снабжения;
- сокращение затрат на закупку материалов и содержание складского комплекса;
- организация слаженной работы сотрудников из разных подразделений;
- увеличение оборачиваемости складских запасов.

Эффективное управление материальными потоками осуществляется с помощью набора функциональных возможностей, приведенных в табл. 1.

Таблица 1

Функциональные возможности модуля SAP MM

Функционал	Основные возможности функционала
Планирование потребности в материалах	Контроль объема имеющихся запасов
	Автоматическое создание заказов для закупки и производства
Создание заказов на поставку	Возможность фиксации информации об услуге или товаре закупки, условиях стоимости позиции, сроках и условиях поставки
	Заказ на поставку может быть создан автоматически или вручную
Организация закупочных процессов	Возможность выбора поставщиков материалов и услуг, контроля статуса заказов и отслеживания платежей. Существует система напоминаний или уведомлений, которая оповещает партнеров о неуплаченных позициях заказов на поставку
Контроль поступлений и перемещений объектов на складе	Информация по реализованным проводкам отражается автоматически в бухгалтерском учете, когда будут автоматически сформированы документы материалов и бухгалтерские документы
	Есть возможность реализовать проводку переносая, при условии, что если перенос будет включать внутренние перемещения объектов между складами организации, они также будут отображаться в отчетности
Отпуск материалов	Существует возможность формирования резервирования отпуска материла для разных счетов (кнтировок), как в ручном режиме, так и в автоматическом
Контроль счетов и оценка запасов	Контроль корректности расчетов легко осуществим благодаря бухгалтерским документам, которые автоматически создаются после проведения счетов-фактуры в системе. Мгновенное и автоматическое обновлении счетов в финансовой бухгалтерии позволяет выполнять оценку запасов материалов на предприятии
Проведение инвентаризации и переоценки	Существует несколько видов инвентаризации в SAP MM: непрерывная, периодическая, выборочная и на определенную дату. Возможно изменение стоимости запасов материалов без изменения количества при проведении переоценки

1.2. Идентификация материально-производственных запасов в системе SAP MM

Идентификация материально-производственных запасов (МПЗ) средствами SAP MM позволяет компании хранить всю информацию о конкретном материале или продукте, является центральным источником информации для материала, относящегося к нескольким областям применения (закупки, хранение, учет, продажи и т. д.), в зависимости от типа материала.

Основные данные по материалам однозначно идентифицируются номером материала, который может быть внутренним или внешним. Внутренние номера автоматически генерируются системой SAP, а внешние номера могут быть явно выбраны пользователями.

Учет основных данных по МПЗ ведется на уровне предприятия, а учетная информация отражается на уровне оценки. Оценкой в данной ситуации считается уровень организационной

структуры, на котором используется данная ассортиментная единица рассматриваемого материала. Учетная информация должна содержать следующую информацию, показанную в табл. 2.

Таблица 2

Основные информационные характеристики, используемые при учете материально-производственных запасов

Информационные характеристики	Описание характеристик
Тип материала	Соотносит материал группе материалов (например, сырье или запасные части)
	Определяет некоторые атрибуты материала и выполняет важные функции управления Определяет способ оценки – по стандартной цене или по скользящей средней цене
Сектор промышленности	Определяет отрасль промышленности, к которой относится материал
Класс оценки	Позволяет группировать различные материалы с похожими свойствами с точки зрения финансовой оценки, чтобы избежать управления отдельным счетом запаса для каждого отдельного материала
Виды оценки	Используются для раздельной оценки. По мере необходимости для каждого вида оценки будет вестись отдельное бухгалтерское представление основной записи материала

2. Представление основных данных по материально-производственным запасам

Основная запись, характеризующая МПЗ, содержит информацию обо всех МПЗ которые компания закупает или производит, хранит и продает.

Основная запись МПЗ используется всеми компонентами в системе логистики SAP. Интеграция всех данных материала в единый объект базы данных исключает избыточное хранение данных.

Представление основных данных показано на примере идентификации LED TV 40 как готовой продукции (рис 1).

Basic Views: PC (шт) – это единица измерения, в которой осуществляется управление запасами материала.

Material Group: используется для группировки нескольких материалов или услуг с одинаковыми атрибутами. Каждый материал или услуга соотносится с определенной группой материалов.

General Item category group: Группа типов позиций NORM определяется для материалов, хранящихся на складе; группа DIEN для услуг и нештатного материала. Данный индикатор влияет на продажи и обработку перемещения запаса.

Weight Unit: KG – единица измерения веса продукта.

Gross weight: 48, 845 – вес брутто 1 единицы продукта в единице веса.

Net weight: 44, 623 – масса нетто 1 единицы продукта в единице веса [7–8].

На следующем этапе отражается классификационный вид. Материалы/готовая продукция могут быть классифицированы для последующего поиска по классу, характеристикам партии и т. д. (рис. 2).

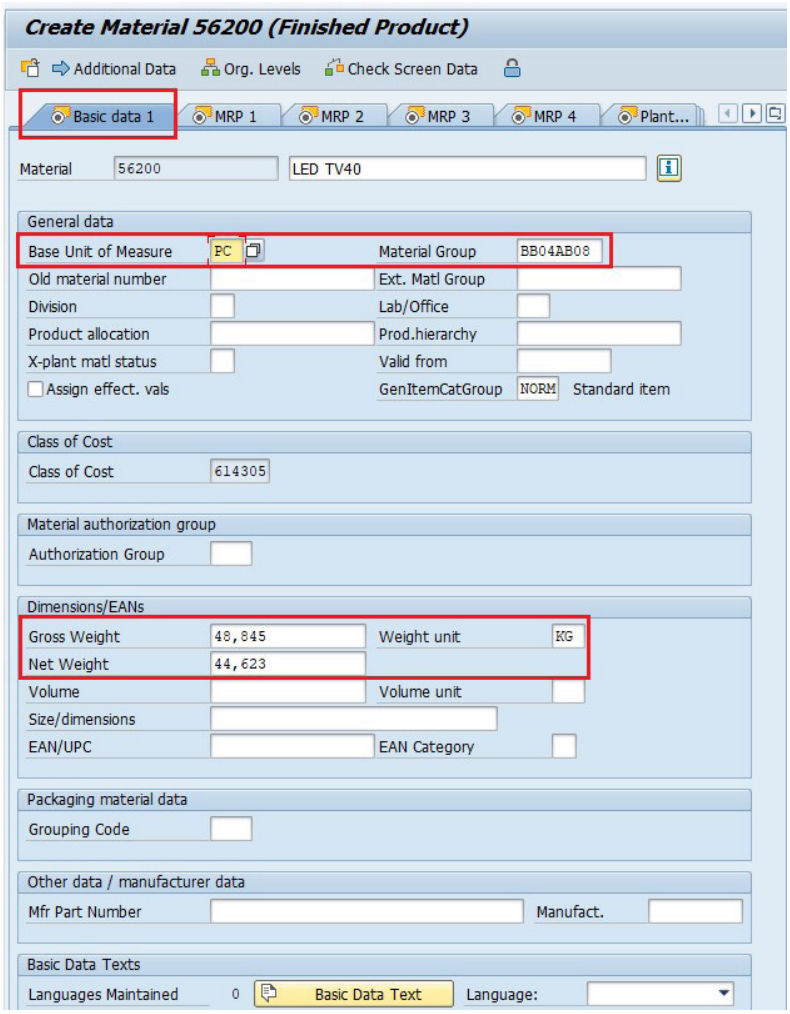


Рис. 1. Ввод основных данных SAP MM

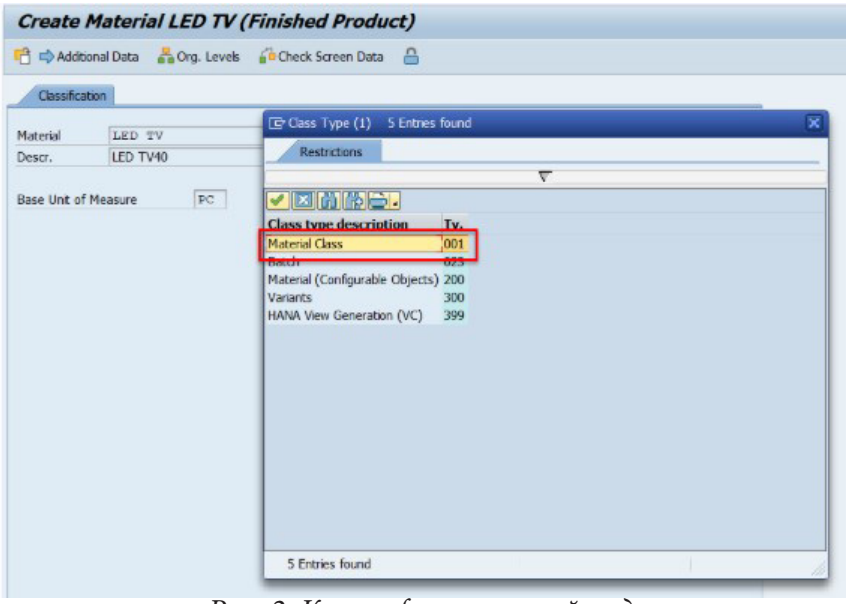


Рис. 2. Классификационный вид

Material Class: при выборе класса материала – 001 осуществляется назначение материала на данный тип класса. Добавленный класс под названием ZMATERIAL может состоять из двух характеристик: ZColora (цвет) и ZDimension (номер) (рис. 3) [8].

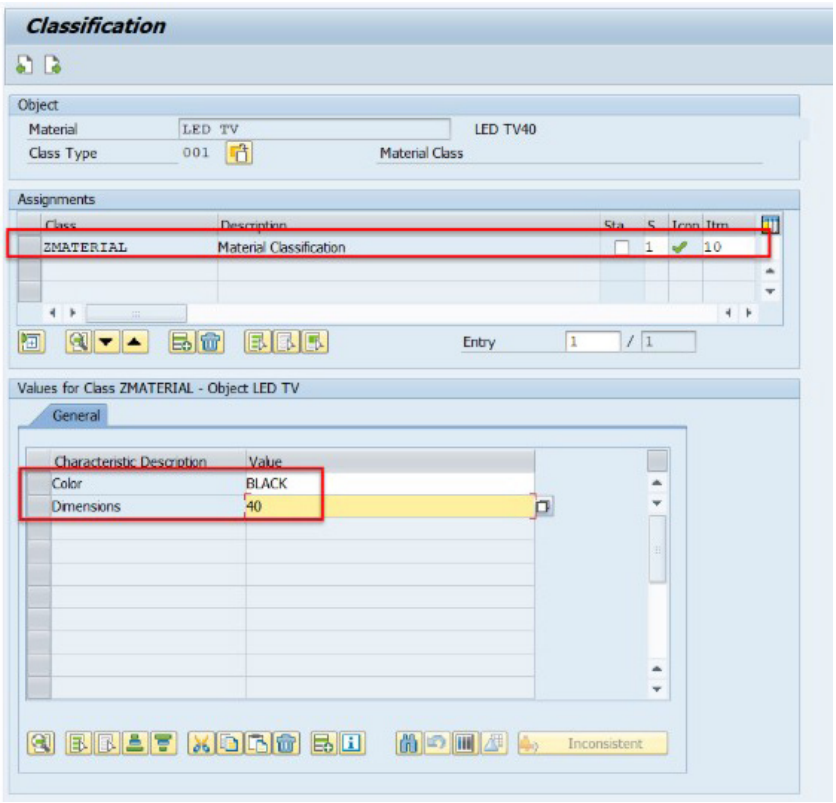


Рис. 3. Расширение классификационного вида

Рассмотрим вкладку Sales Organization Data 1. При выборе этого представления мы видим инструмент выбора различных организационных уровней. Он дает возможность конкретизации выпускающего предприятия, сбытовой организации и канала сбыта нашей продукции рис. 4.

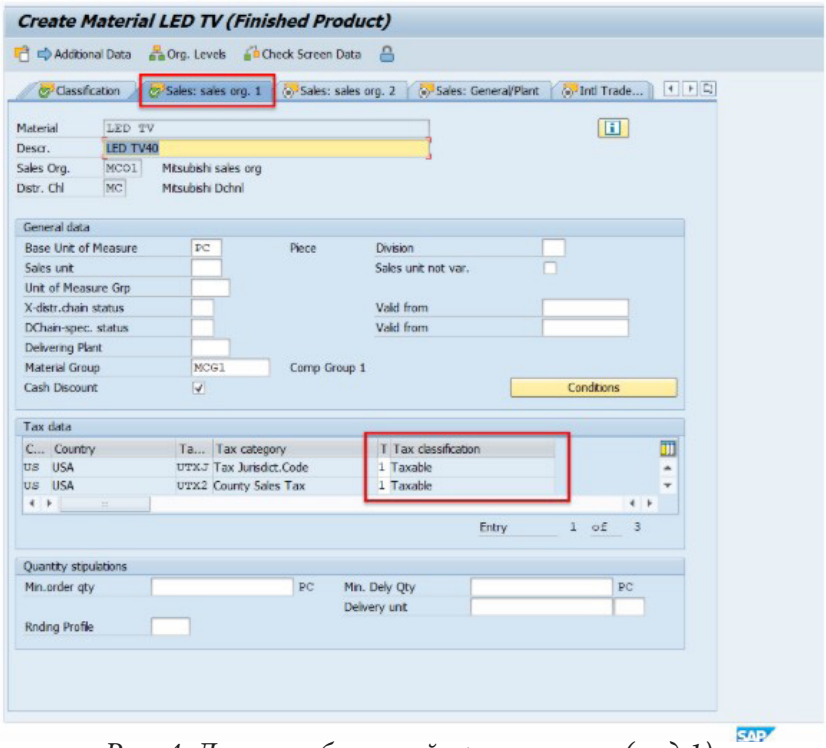


Рис. 4. Данные сбытовой организации (вид 1)

На вкладке Sales Organization Data 2 отображаются:

Account assignment group: данная характеристика определяет требования к учету материала. Для разных типов материалов требуются разные требования к учету (готовая продукция, торговые товары, услуги не учитываются одинаково). Это своеобразная точка интеграции с модулями финансов и контроллинга [8].

Item category group: позволяет группировать связанные категории товаров в торговом документе. Он присваивается основной записи материала через виды материала и связывается с модулем сбыт с помощью правила определения типа позиции (рис. 5).

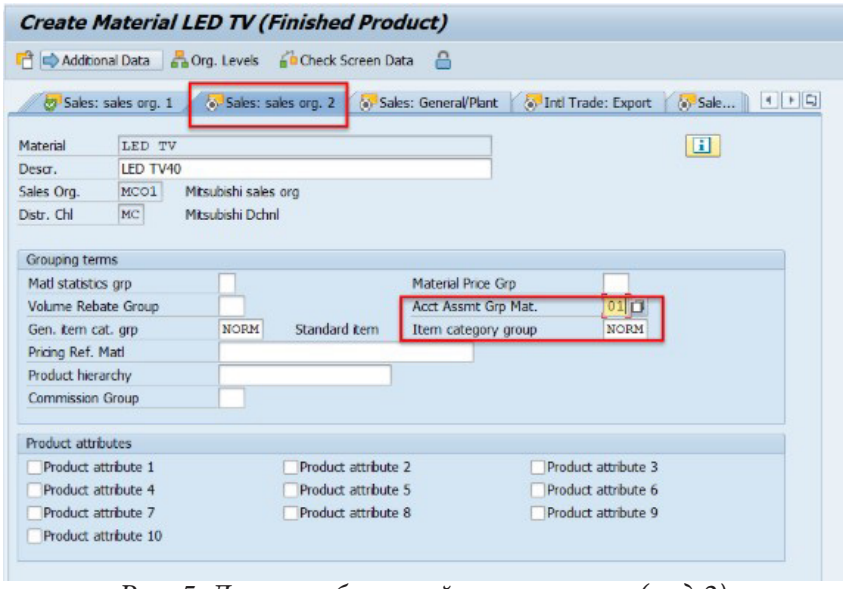


Рис. 5. Данные сбытовой организации (вид 2)

Вкладка Sales General / Plant View:

Availability check: используется для определения метода проверки наличия на складе. Эти методы определяются командой по настройке и могут включать запасы на складах, также могут быть настроены на включение необходимого объема материалов/продукции в технологические и производственные заказы, плановые заказы и т. д.

Loading group: чрезвычайно важное поле, поскольку оно используется при определении пункта отгрузки.

Transportation group: это способ группировки материалов с одинаковыми транспортными потребностями. Он используется для планирования маршрутов в заказах на продажу и поставках.

Material Group – Packaging Materials: это поле позволяет сгруппировать материалы с аналогичными требованиями к упаковке (рис. 6).

В условиях программного продукта компании SAP осуществляется взаимодействие и внедрение модуля MM с другими модулями, а именно:

Финансы и Контроллинг (Finance and Controlling, FI–CO) – заказы на поставку, счета поставщиков, платежи поставщикам, перемещения запасов, проверка качества, инвентаризация запасов, несовпадения в запасах, программа платежей и т. д. [1]. Модуль FI связан с модулем MM, потому что каждая операция, выполняемая в модуле MM, такая как получение материала и получение счета-фактуры, напрямую влияет на финансовую операцию организации.

Продажи и Дистрибуция (Sales and Distribution, SD) – проверка наличия, расписание поставок, проверка кредита, поставки материалов, перемещения запасов между разными участками производства, определение материалов, исключение материалов, замена материалов, пункты повторного заказа, возврат.

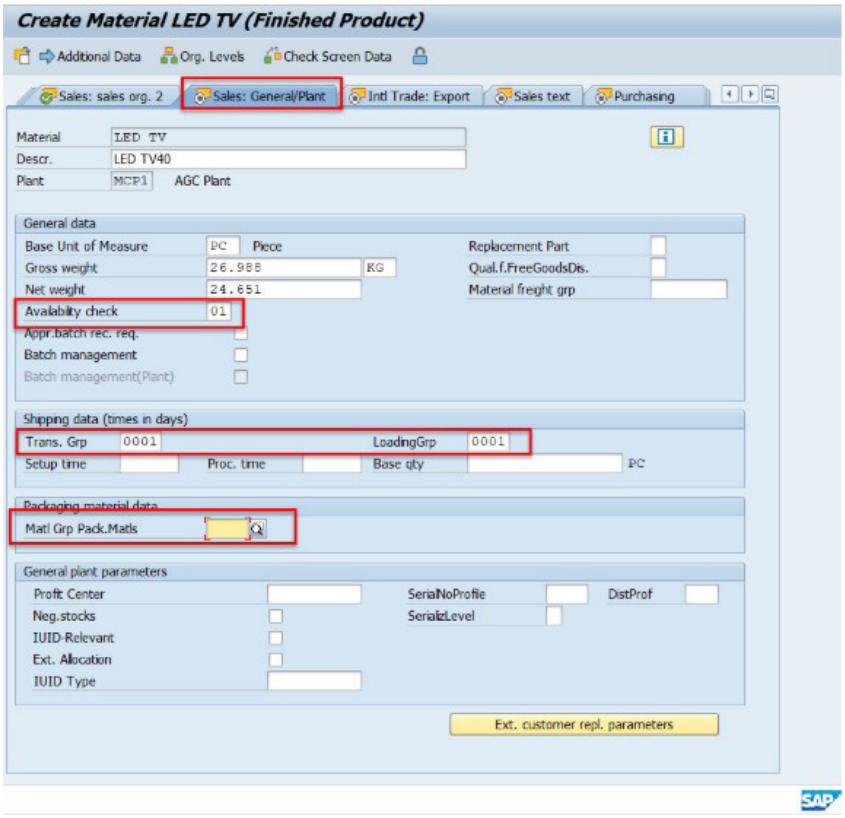


Рис. 6. Продажи: общие/завода

Планирование производства (Production Planning, PP) – MM занимается заготовкой МПЗ на основе требуемой продукции, поэтому он связан с модулем PP. Он интегрирован в такие области, как ППМ, поступление и выпуск материала по производственному заказу. Планирование потребности в материалах основывается на запасах, ожидаемых поступлениях, ожидаемых проблемах. Генерирование плановых заказов или заявок, которые могут быть преобразованы в заказ / контракт на покупку. Управление запасами отвечает за подготовку ресурсов, необходимых для производственных заказов. Поступление готовой продукции на склад проводится в Управлении запасами.

Управление качеством (Quality Management, QM) – запись информации о качестве, решения об использовании и т. д.

Обслуживание и Ремонт (Plant Maintenance, PM) – заказы на поставку, управление внешними услугами и т. д.

Заключение

При рассмотрении функциональных возможностей модуля SAP Materials Management (MM), становится очевидным, что он поддерживает все аспекты управления материальными ресурсами (планирование, контроль и т.д.), является частью области логистики и помогает управлять закупочной деятельностью организации.

Модуль SAP MM позволяет объединить основные направления закупочной логистики, – продажи и дистрибуция, планирование производства, производственной логистики – техническое обслуживание завода, системы проектов, складской логистики – управление складами.

Литература

1. *Чалдаева, Л. А.* Экономика предприятия : учебник для бакалавров / Л. А. Чалдаева. – 3-е изд., перераб. и доп. – Москва : Юрайт, 2013. – 411 с.

2. *Вейс, В.* Разработка приложений SAP R/3 на языке ABAP/4. / В. Вейс. – Москва : Инфра-М, 2010. – 617 с.

3. *Савкин М. И.* Автоматизация процессов в SAP BusinessObjects Planning / М. И. Савкин. – Москва : Проспект, 2009. – 733 с.

4. *Освальд, Г.* SAP Enterprise Support / Г. Освальд, У. Хоммель. – Москва : Инфра-М, 2010. – 547 с.

5. *Рид, Д.* Настольная книга SAP-консультанта / Д. Рид, М. Доан. - Москва : Эксмо, 2009. – 621 с.

6. Управление материальными потоками SAP MM. – Режим доступа: http://www.itctg.ru/solutions/sap-mm-material. – (Дата обращения: 15.04.2020).

7. SAP S4 HANA MM (Material Management). – Режим доступа: https://www.gauravconsulting.com/sap-s4-hana-mm-training-course. – (Дата обращения: 18.04.2020).

8. SAP S/4 HANA Material Master. – Режим доступа: https://blogs.sap.com/2019/08/12/sap-s4-hana-material-master/. – (Дата обращения: 20.04.2020).

Хоф Светлана Алексеевна – магистрант 1-го года обучения кафедры математических методов исследования операций Воронежского государственного университета.

E-mail: svetahof@gmail.com

Булгакова Ирина Николаевна (научный руководитель) – д-р экон. наук, доцент, доцент кафедры математических методов исследования операций Воронежского государственного университета. E-mail: bulgakova-i-n@yandex.ru

УДК 004.451.46

БАЛАНСИРОВКА НАГРУЗКИ КООПЕРАТИВНОЙ МНОГОЗАДАЧНОЙ СИСТЕМЫ В МНОГОЯДЕРНОЙ СРЕДЕ

А. В. Цыганкова

Воронежский государственный университет

Введение

Формирование современной СУБД для работы на многоядерном оборудовании требует особой софтверной платформы разработки. Анализ состояния и тенденций развития технологий СУБД показывает очевидный прорыв технологий, базирующихся на неблокирующих алгоритмах в работе с данными в памяти. Однако только общее использование единого неблокирующего подхода позволит достичь по-настоящему выдающихся результатов. Специализированные СУБД «всё в памяти» демонстрируют хорошие результаты, но разработанные методики не удается совместить в архитектуре классической СУБД с вытеснением данных. Требуется использование медленной синхронизации, плохо совмещаемой с неблокирующими подходами.

1. Платформа разработки

1.1. Общее описание

Для решения задачи построения параллельных вычислений необходим общесистемный подход к переходу от быстрых алгоритмов синхронизации к медленным без потери свойств быстрых алгоритмов. Исследования существующих систем показывают, что для этого в первую очередь необходим новый, альтернативный набор примитивов синхронизации и методика комбинирования быстрых и медленных алгоритмов.

Для сохранения эффективности функционирования любых неблокирующих алгоритмов, необходимо соблюсти простое правило: части неблокирующих алгоритмов, подразумевающие возможную конкуренцию, должны исполняться на ЦПУ непрерывно, без передачи управления другому потоку или задаче. Безусловно, обеспечить полную гарантию запрета прерываний можно только из ядра ОС, но задача минимум – обеспечить наилучшие условия для минимизации возможных прерываний. Таким образом, были сформулированы требования к платформе разработки, которая позволит формализовать и стандартизировать подходы совместного эффективного использования ранее несовместимых алгоритмов и скрыть их сложность. Платформа разработки обеспечивает изоляцию кода платформы от использующего её кода и позволяет минимизировать затраты по портированию полученного решения под разные ОС.

1.2. Архитектурные требования

Архитектура платформы должна удовлетворять ряду условий:

• Количество активных потоков не должно превышать количества свободных ядер в ОС, отведённых для платформы.

• Запрещено переключать контекст исполнения текущей задачи: выполнять любые системные вызовы и производить управление памятью системными средствами, при исполнении неблокирующего алгоритма по конкурентному доступу к данным.

• Следует избегать длительных вычислений в операциях, построенных на неблокирующих алгоритмах, которые могут увеличить время вынужденной быстрой синхронизации в таких алгоритмах.

• Платформа подразумевает повсеместное использование неблокирующих алгоритмов и запрещает использование примитивов синхронизации, построенных на принципах взаимного исключения (пример: Spin, Mutex).

• Платформу должна быть обеспечена инструментами отладки собственных примитивов.

• Решение проблемы конкурентного доступа к памяти имеет наивысший приоритет.

• Платформа должна обладать технологичностью, в том смысле, что она должна позволять трансформировать известные методики решения стандартных задач под условия платформы.

1.3. Модель параллельных вычислений в СУБД, многозадачность

В СУБД известны разные модели многозадачности, в представленной архитектуре реализуется кооперативная многозадачность. Такой выбор обусловливается множеством причин и требований:

• Контекст кооперативной задачи может быть значительно минимизирован под требования СУБД относительно системных примитивов thread, process.

• СУБД реализует собственные примитивы синхронизации, которые требуют контролируемого переключения между задачами, полный контроль легче осуществлять в кооперативной многозадачности. Кооперативная многозадачность позволяет строить примитивы синхронизации без лишнего переключения контекста в планировщик (или ядро ОС).

• Для уменьшения алгоритмической сложности активно используются синхронные интерфейсы, построенные на основе обобщенных примитивов кооперативной многозадачности. Таким образом, скрываются детали реализации асинхронного взаимодействия с ОС, управления памятью и взаимодействия между компонентами внутри системы.

• Для возможности повсеместного использования неблокирующих алгоритмов необходим обобщённый контроль доступа к памяти. Имеется ввиду способ рекламации памяти типа: Read Copy Update, Epoch Based Reclamation (EBR). Такие механизмы хорошо интегрируются посредством кооперативной многозадачности.

• Контролируемый процесс переключения контекста задачи позволяет реализовать комбинированный контроль доступа к памяти. Для быстрых, неблокирующих алгоритмов, используем EBR секции, а при необходимости медленной синхронизации, переключаем защиту на альтернативный способ – медленная синхронизация с досрочным выходом из секции EBR. Далее, после исполнения условий синхронизации, заново активируем EBR секцию и продолжаем работу неблокирующих алгоритмов.

• Объекты синхронизации строятся на возможности исключения обмена сообщениями, или, если обмен сообщениями необходим, то асинхронность обмена позволяет скрыть или уменьшить время задержки доставки сообщения.

• Для реализации возможности параллельных вычислений, система запускает набор исполняющих очередей по количеству ЦПУ, с расположением каждого контекста исполняющей очереди в выделенном потоке. В рамках исполняющей очереди, исполняется локальный алгоритм планирования задач. Код балансирования нагрузки между разными исполняющими очередями интегрирован в базовые функции планировщика.

• Решения проблем многозадачности имеют аналогии с современными решениям данной проблемы в ядрах Linux, FreeBSD. Принципиальное отличие в том, что многозадачность кооперативная и полностью управляется из пользовательского пространства. Отсутствие необхо-

димости переключения контекста между ядром и пользовательским пространством активно используется для оптимизации операций в системе.

В данной статье, рассмотрим одну из проблем по организации многозадачности в рассматриваемой платформе разработки. А именно – проблема планирования задач и балансировки вычислительной нагрузки в среде многоядерного оборудования.

2. Механизм планирования исполнения задач

2.1. Планировщик

Механизм планирования исполнения задач включает в себя набор очередей и правила для балансировки нагрузки очередей, где взаимодействие основано на своих «легковесных» (относительно системных) механизмах синхронизации. Дополнительная оптимизация процесса достигается за счёт исполнения планировщика и задачи в пользовательском контексте. Текущее исследование посвящено алгоритму балансировки нагрузки системы, где организовано «честное» планирование задач, в зависимости от их приоритетов.

Задачи планировщика:

- эффективно использовать доступные ресурсы системы;
- минимизировать конкуренцию;
- уменьшить вычислительную сложность;
- максимизировать использование кэша процессора;
- обеспечить равномерное распределение нагрузки исполнения задач.

Для достижения наибольшей выгоды, планировщик должен эффективно использовать все ресурсы системы. Другими словами, необходимо обеспечить равномерную работу всех ядер ЦПУ. Планировщик ассоциирует каждую очередь с конкретным ядром. Откуда очевидно, что число рабочих нитей не должно превышать общее число ядер ЦПУ системы.

В процедуре планировщика не используются неблокирующие алгоритмы, поскольку эксплуатация неблокирующих алгоритмов усложнит вычисления. Поэтому применяются классические варианты на спинах. Атомарные операции служат для выяснения условий необходимости работы блокировок. Текущие измерения показывают, что выбранный вариант реализации удовлетворяет современному оборудованию, то есть нет проблем с конкуренцией.

Базовый алгоритм по планированию включает в себя балансировку нагрузки системы (рис. 1).

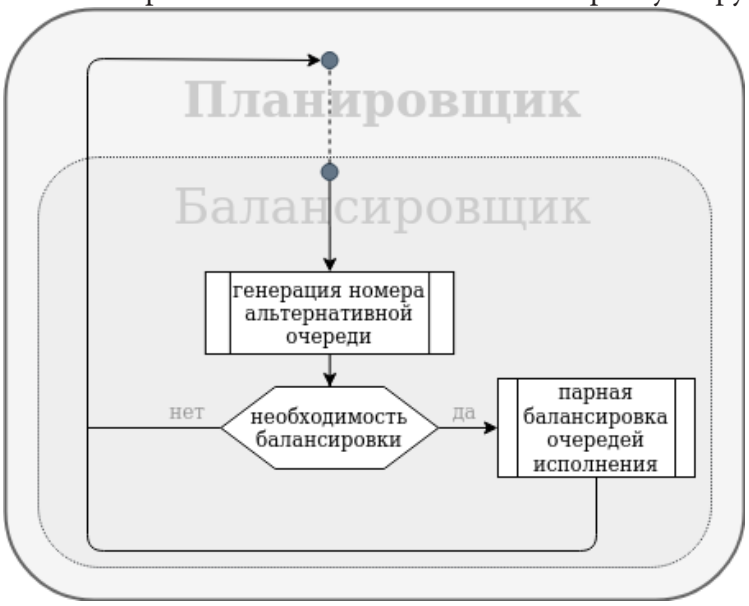


Рис. 1

Процедура перебалансировки запускается не на каждой итерации цикла планирования, а при соблюдении ряда условий, что уменьшает вычислительную сложность.

Очередь исполнения содержит упорядоченный по дедлайну список задач. Для хранения элементов был реализован контейнер скиплист. Данный выбор обусловлен тем, что скиплист имеет сложность вставки равную O(log N), как и в дереве, но извлечение наименьшего элемента имеет сложность O(1), в отличие от дерева, где она равняется O(log N). Задача попадает в очередь исполнения на основании своего параметра – дедлайн. Расчёт дедлайна зависит от приоритета задачи и от момента старта.

$$deadline = startMoment(time, priopityPolicy) + priority^{*}timeSlice,$$

где:

- priority – приоритет задачи – значение, заданное при старте, где priority = 1–2^16;
- timeSlice – квант времени;
- priorityPolicy – политика приоритета – значение на «условной временной шкале», с которого начинается расчёт дедлайна. Значения временной шкалы определены в диапазоне от 0 до (2^64–2^16). Наиболее приоритетные задачи определяются политикой со значением 0. Задачи, которые не имеют необходимости в срочном исполнении, имеют политику «текущее время», условно можно считать, что это середина диапазона. Так же существует ряд задач, которые должны быть выполнены, но не имеет значения, в какой момент времени – им присваивается политика со значением (2^64–2^16).

Таким образом, алгоритм планирования нацелен на исполнение в первую очередь тех задач, значение дедлайна у которых наименьшее.

2.2. Балансировка нагрузки

Работа планировщика представляет собой цикличный процесс. Набор очередей, в момент времени, ассоциирует каждую нить с конкретным ядром ЦПУ. На каждой итерации цикла работает проверка на соответствие критериям балансировки. В случае, если все условия соблюдены, вызывается процедура перераспределения нагрузки между двумя очередями. Также необходимо отметить, перебалансировка может происходить в двух направлениях, то есть, если основная очередь бОльшей длины, то задачи будут мигрировать из основной очереди во вспомогательную, и наоборот – если основная очередь содержит меньше задач, то она будет принимать элементы из бОльшей очереди. Таким образом, необходимо корректно выбрать альтернативную очередь исполнения для распределения задач.

Большая часть работы происходит без какой-либо конкуренции, необходимость балансировки проверяется на основе атомарной операции чтения дедлайна соседней очереди исполнения, и попытки захвата очереди на балансировку без ожидания.

Процесс балансировки запускается в том случае, если выполняются условия:

- главное условие – большая разница между дедлайнами голов очередей, проверяется без использования блокировок;
- число задач, хотя бы в одной очереди, больше одной;
- захвачена блокировка второй очереди.

```
/* load balance condition */
uint32_t other_entries = atomic_load_explicit(&other_rq->skiplist.entries,
                         memory_order_relaxed);
uint32_t self_entries = atomic_load_explicit(&self->skiplist.entries, memory_order_relaxed);
uint32_t worthy = (other_entries > 1 || self_entries > 1)
                  && __WORTHY_DIFF(self->shared) < (next_key > best_key
                                  ? next_key - best_key
                                  : best_key - next_key);
```

Устройство памяти современного компьютера, применимое к мультипроцессорным системам, имеет иерархичную топологию – так называемый «неравномерный доступ к памяти» или Non–Uniform Memory Architecture(NUMA). NUMA узел определяет набор ядер, для которых доступ к памяти данного узла имеет меньшую цену, чем доступ к памяти удалённого узла. Так же, миграция задачи на другое ядро, может приводить к увеличению времени повторного доступа к участкам памяти, из-за обращения к удаленному кешу, располагаемому в другом NUMA узле. Поэтому перемещение контекста исполнения задач между NUMA узлами – наиболее дорогая операция, относительно перемещений внутри NUMA узла.

Для компенсации задержек при перемещении задач, для минимизации «дорогих» миграций из одного NUMA узла и обратно, необходимо учитывать, при балансировке нагрузки, расположение ядер ЦПУ. Другими словами, для достижения наибольшей эффективности, необходимо увеличить вероятность балансировки между очередями исполнения одного NUMA узла. Чтобы обеспечить балансировку нагрузки преимущественно с очередями исполнения из одного узла, планировщик использует процедуру «рандомной» генерации очереди.

Цель данного алгоритма – вернуть номер альтернативной исполняющей очереди для парной балансирования нагрузки задач. Учитывая требования планировщика – минимизация конкуренции доступа к памяти, уменьшение вычислительной сложности, становится очевидно, что выбор альтернативной очереди может зависеть только от топологии системы. Получение какой-либо информации о состоянии системы в момент времени – узкое место, которое противоречит требованиям планирования.

Алгоритм генерации очереди основывается на распределении шансов балансировки. Шансы, в свою очередь, формируются исходя из организации памяти – учитывается число NUMA узлов, число гипертредных ядер и число логических ядер в одном NUMA узле. Сумма всех вышеперечисленных значений формирует диапазон распределения случайной величины. Таким образом, дальнейшая генерация рандомного числа и сопоставление с диапазоном распределения, определяет область, где будет выбрана альтернативная исполняющая очередь для парной балансировки нагрузки. В результате, остаётся случайно определить номер очереди из уже известной области памяти.

Рассмотренный алгоритм даёт возможность:

• выполнять балансировку с учётом топологии системы;

• избежать остро-неравномерной загруженности системы – благодаря рандомной генерации номера очереди, балансировка всегда выполняется для разных пар очередей исполнения. Иными словами, при загруженности на каком-либо участке, нагрузка не будет распределена по заранее определённому сценарию перебора очередей, где с высокой вероятностью может быть не захвачена блокировка из-за исполнения «тяжёлой» задачи с высоким приоритетом, и, как следствие, будет организована высокая загруженность отдельной области.

Заключение

Задача балансировки нагрузки возникла при разработке особой софтверной платформы для работы на многоядерном оборудовании. Архитектура платформы базируется на принципах кооперативной многозадачности, исходя из этого имеется возможность реализации неблокирующих алгоритмов, лёгких примитивов синхронизации, также достигнут обобщенный контроль доступа к памяти и существует контролируемый процесс переключения контекста задачи. Планирование работы очередей задач направлено на бесконкурентную работу, а вычислительная сложность условий необходимости балансировки сведена к минимуму.

Таким образом, исследование, рассмотренное в данной статье, направленное на поиск наиболее оптимального и дешёвого способа балансирования нагрузки задач исполнения ядра СУБД, позволило:

- обеспечить полную бесконкурентную работу планировщика;
- реализовать честное планирование задач;
- равномерно распределить нагрузку на систему;
- обеспечить минимальную конкуренцию в процессе балансировки;
- минимизировать вычислительную сложность;
- наиболее эффективно использовать ресурсы системы.

В результате рассмотренный алгоритм балансировки нагрузки признан достаточным для ядра СУБД, поскольку перечисленные выше требования для платформы удовлетворены.

Литература

1. Национальная библиотека им. Н. Э. Баумана: Многозадачность (Операционные Системы) – Режим доступа: https://ru.bmstu.wiki/Многозадачность_(Операционные_Системы).html (Дата обращения: 23.04.2020).

2. Programming land: Многозадачные операционные системы. – Режим доступа: http://programming-lang.com/html/visual_studio.net/glava12/index5.htm (Дата обращения: 23.04.2020).

3. OpenNET: МНОГОЗАДАЧНОСТЬ – Режим доступа: https://www.opennet.ru/docs/RUS/zlp/006.html (Дата обращения: 23.04.2020).

4. Процессы в Windows и потоковая многозадачность. – Режим доступа: http://www.scritub.com/limba/rusa/Windows11518232111.php (Дата обращения: 23.04.2020).

Цыганкова Анастасия Владимировна – магистрант 2-го года обучения кафедры вычислительной математики и прикладных информационных технологий Воронежского государственного университета. E-mail: TsyganckovaAnastasia@yandex.ru

Борисенков Дмитрий Васильевич (научный руководитель) – канд. техн. наук, доцент кафедры Математического Обеспечения ЭВМ Воронежского государственного университета. E-mail: xuser@relex.ru

УДК 004.622

РАЗРАБОТКА МОДУЛЕЙ ИНТЕГРАЦИИ СИСТЕМЫ 1С:ПРЕДПРИЯТИЕ С ИНФОРМАЦИОННО-РЕЙТИНГОВЫМИ СИСТЕМАМИ В СЕТИ ИНТЕРНЕТ

Е. М. Шабунин

Воронежский государственный университет

Введение

Успех в работе предприятия во многом зависит от уровня знаний и профессионализма наемных работников. Качественный продукт и успешный бренд, безусловно, важны, но главную роль все равно играют талантливые сотрудники, которые могут грамотно управлять и продажами, и брендами, и производством. Любой продукт можно скопировать, любую технологию воспроизвести. Единственное, что точно нельзя повторить, это людей, которые обеспечат движение бизнеса вперед. Именно таланты влияют на то, каким будет бизнес в будущем. Поэтому сегодня работодатели как никогда сфокусированы на развитии потенциала, который уже есть внутри компании в лице собственных сотрудников, а также на привлечении необходимых талантов извне. Для этого на рынке существует много продуктов для поиска необходимых кадров, например ресурс hh.ru. Данный ресурс один из немногих имеет открытую API – документацию для интеграции с другими продуктами.

В свою очередь API hh.ru возможно интегрировать с 1С:Предприятие для необходимого функционала HR – отдела каждой компании, работающей с 1С. Данная интеграция значительно оптимизирует и облегчает работу сотрудников HR – отдела и позволяет отбирать не только сотрудников в данный момент, но и вести анализ вакансий и резюме на данном ресурсе.

1. Конфигурация для интеграции с hh.ru

Для реализации данной интеграции была выбрана платформа 1С:Предприятие, средство разработки 1С:EDT[1], ресурс GitHub для разработки с помощью системы контроля версий Git и открытая документация API – интеграции ресурса hh.ru. Система контроля версий Git была выбрана для упрощения разработки автономной конфигурации на основе 1С:Предприятие.

Преимущество Git заключается в том, что возможна групповая разработка в одной конфигурации, что невозможно в стандартном средстве разработки, «Конфигураторе». Принцип Git заключается в записи изменений в конфигурации в так называемый «коммит» – запись о том, что именно было доработано в конфигурации, что позволяет в любой момент вернуться к любому «коммиту» без значительных затрат ресурсов. Например, если проводить разработку без помощи Git, то вся информация о доработках, после обновления конфигурации исчезает бесследно. Взаимодействие с 1С и Git происходит с помощью средства разработки 1С:EDT, созданной компанией 1С для значительного упрощения взаимодействия разработчика и конфигурации 1С. По сравнению со стандартным средством разработки, «Конфигуратор», 1С:EDT имеет ряд инструментов для упрощения работы с конфигурацией 1С, в том числе Git.

Была создана пустая конфигурация в 1С:Предприятие и внедрена часть БСП (Библиотека стандартных подсистем). Инструментарий разработчика «1С:Библиотека стандартных подсистем» (БСП) предоставляет набор универсальных функциональных подсистем, готовые разделы для пользовательской документации и технологию для разработки прикладных решений на платформе «1С:Предприятие». С применением БСП становится возможной быстрая разработка новых конфигураций с уже готовой базовой функциональностью, а также включение готовых функциональных блоков в существующие конфигурации. Использование БСП при

разработке прикладных решений на платформе «1С:Предприятие» позволяет также достичь большей стандартизации конфигураций и уменьшить время на изучение и внедрение прикладных решений за счет их унификации по набору используемых стандартных подсистем. Входящие в БСП подсистемы охватывают такие области, как администрирование информационной базы, администрирование пользователей программы, настройка доступа к данным информационной базы, ведение различной нормативно-справочной информации (адресный классификатор, курсы валют, календарные графики и др.). В нашем случае необходимы были как минимум подсистемы администрирования, настройка доступа к данным, а также модули работы с адресными данными. Далее созданы справочники: регионы, специализации, метро, языки, отрасли компании, учебные заведения, регионы (рис. 1). Также были созданы документы для записи вакансий и резюме для дальнейшего анализа данных.

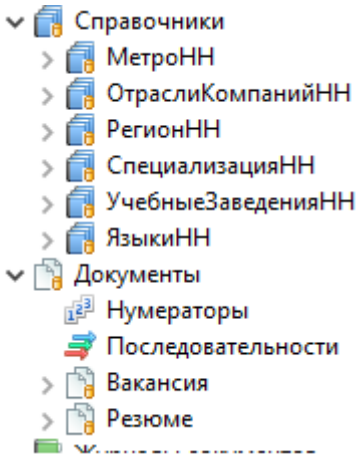


Рис. 1. Общая схема конфигурации

В общих модулях прописан «Коннектор HTTP» для подключения к серверам hh.ru, а также модули для получения и преобразования данных с серверов hh.ru в 1С.

После получения необходимых данных, мы можем взаимодействовать с данной информацией и проводить анализы по заработной плате у других компаний с помощью вакансий, а также уровень соискателей на заданном этапе для анализа рынка труда. В результате дальнейших доработок планируется создать несколько отчетов, анализирующих данные по вакансиям и резюме за определенный промежуток времени, задаваемый пользователем. Данный отчет может помочь дать оценку рынка по многим параметрам. Например, возраст соискателей, зарегистрированных на сайте, может показать средний возраст рабочего населения в России, который ищет работу. Также в общей выкладке можно увидеть, например, насколько готовы студенты университетов, обучающиеся или только завершившие университет, работать по направлению своего образования.

Заключение

Данная конфигурация является отдельной, однако её возможно доработать и интегрировать в любые продукты 1С: Зарплата и управление персоналом, Комплексная Автоматизация, ERP. Преимущество отдельной конфигурации заключается в том, что если её развернуть на отдельной базе, возможна работа только тех пользователей, которым необходимы данные с сайта hh.ru и которые работают в HR – отделе, без создания отдельный ролей и подсистем для интеграции с другими продуктами 1С. Однако, интеграция с другими продуктами 1С возможна. Для этого необходимо понять, с каким именно продуктом будет интегрироваться данная конфигурация для доработки обращений к общим модулям в теле справочников и документов.

Литература

1. Документация по 1С:EDT. – https://its.1c.ru/db/edtdoc

2. *Хрусталева, Е. Ю.* Технологии интеграции 1С:Предприятия 8.3 / Е. Ю. Хрусталева. – «1С-Паблишинг» – 2020.

Шабунин Евгений Михайлович – студент 4-го курса кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: kvadra_shabunin@mail.ru

Сафронов В.В. (научный руководитель) – канд. техн. наук, доц., доцент кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: vitolik@bk.ru

УДК 004.942

СИМУЛЯЦИЯ СОЦИУМА. ПРОБЛЕМЫ И ПУТИ ИХ РЕШЕНИЯ

А. Г. Шенгелия

Воронежский государственный университет

Введение

Все люди взаимодействуют друг с другом. Общаются, делятся эмоциями, выражают своё мнение и так далее. И для многих прикладных задач часто встаёт вопрос о том, как можно спрогнозировать взаимодействие огромного числа людей между собой. Люди живут в социуме.

Общество, *или социум* – человеческая общность, специфику которой представляют отношения людей между собой, их формы взаимодействия и объединения [1].

1. Постановка задачи

В современном мире симуляция социума – проблема, решением которой занимаются по всему миру. Люди пытаются отыскать модели и методы для наиболее качественных предсказаний поведения объектов внутри социума. Это привело к созданию многочисленных моделей, которые помогают создавать социальные симуляции.

Наибольшей проблемой является то, что для таких симуляций необходимо учитывать различное число параметров (зачастую, огромное число параметров) для огромного числа объектов.

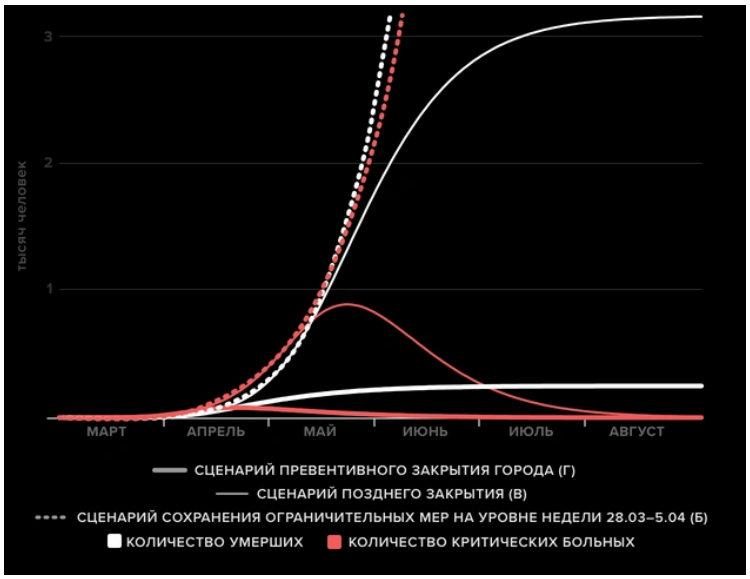


Рис. 1. График, построенный в результате исследования математической модели распространения вируса при различных условиях [2]

Именно по этой причине сегодня не существует моделей, которые на 100 % предсказывали бы поведение социума на сколь угодно значительный промежуток времени вперёд.

Может показаться, что симуляции социумов – незначительная или вовсе бесполезная задача. Однако, это не так.

В современном мире различные модели и методы симуляции социального поведения помогают решать многие проблемы медицины, биологии, социологии, антропологии, экономике и так далее.

Например, биологи могут предсказать при помощи симуляций, как себя будут вести колонии различных существ (бактерий, пчёл и т.д.). Эпидемиологи могут строить различные модели, предсказывающие распространение болезней, и как это может повлиять на социум. Экономисты же такими методами исследуют поведение рынка.

В современном мире уже существует множество способов симуляции общественного поведения. Приведёт несколько примеров.

2. Пути решения

2.1. Игра «Жизнь»

Рассмотрим, например, так называемый клеточный автомат, или, его частный случай, игру «Жизнь». Игра проходит по следующим правилам:

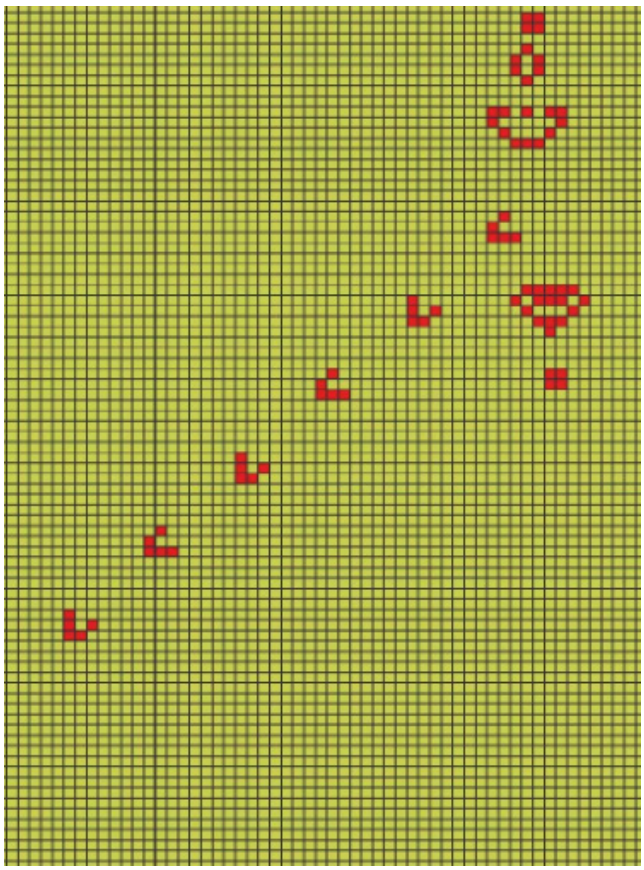


Рис. 2. Поле, на котором происходит игра

• Место действия этой игры – «вселенная» – это размеченная на клетки поверхность или плоскость – безграничная, ограниченная, или замкнутая (в пределе – бесконечная плоскость).

• Каждая клетка на этой поверхности может находиться в двух состояниях: быть «живой» (заполненной) или быть «мёртвой» (пустой). Клетка имеет восемь соседей, окружающих её.

• Распределение живых клеток в начале игры называется первым поколением. Каждое следующее поколение рассчитывается на основе предыдущего по таким правилам:

– в пустой (мёртвой) клетке, рядом с которой ровно три живые клетки, зарождается жизнь;

– если у живой клетки есть две или три живые соседки, то эта клетка продолжает жить; в противном случае, если соседей меньше двух или больше трёх, клетка умирает («от одиночества» или «от перенаселённости»).

• Игра прекращается, если

– на поле не останется ни одной «живой» клетки;

– конфигурация на очередном шаге в точности (без сдвигов и поворотов) повторит себя же на одном из более ранних шагов (складывается периодическая конфигурация);

– при очередном шаге ни одна из клеток не меняет своего состояния (складывается стабильная конфигурация; предыдущее правило, вырожденное до одного шага назад).

• Эти простые правила приводят к огромному разнообразию форм, которые могут возникнуть в игре.

• Игрок не принимает прямого участия в игре, а лишь расставляет или генерирует начальную конфигурацию «живых» клеток, которые затем взаимодействуют согласно правилам уже без его участия (он является наблюдателем).

Игра «Жизнь» является примитивным симулятором поведения социума, однако она не лишена недостатков:

• Малый набор факторов, влияющих на поведение

• Малый набор состояний, в которых объект может находиться (всего 2: «жив» и «мёртв»)

• Высокая предсказуемость, что не позволяет использовать данную модель для реализации симуляции реального социума.

2.2. Рой частиц

Метод роя частиц (МРЧ) – метод численной оптимизации, для использования которого не требуется знать точного градиента оптимизируемой функции.

МРЧ был доказан Кеннеди, Эберхартом и Ши [3] и изначально предназначался для имитации социального поведения. Алгоритм был упрощён, и было замечено, что он пригоден для выполнения оптимизации. Книга Кеннеди и Эберхарта описывает многие философские аспекты МРЧ и так называемого роевого интеллекта. Обширное исследование приложений МРЧ сделано Поли. МРЧ оптимизирует функцию, поддерживая популяцию возможных решений, называемых частицами, и перемещая эти частицы в пространстве решений согласно простой формуле. Перемещения подчиняются принципу наилучшего найденного в этом пространстве положения, которое постоянно изменяется при нахождении частицами более выгодных положений.

Метод невероятно хорош для решения многих оптимизационных задач, но он так же имеет некоторые недостатки:

• Выбор оптимальных параметров метода роя частиц – тема значительного количества исследовательских работ, что не позволяет создать «идеальной» модификации данного метода, которая подходила бы в большинстве ситуаций.

• Малый набор факторов, влияющих на поведение

• Склонность к «влиянию» (рой всегда стекается туда, где, как ему кажется, оптимум, а социум далеко не всегда себя так ведёт)

Заключение

Принимая во снимание всё вышесказанное, можно сделать вывод, что для решения задачи реализации модели симуляции социума необходимо как минимум решить следующие проблемы:

1. Расширить набор факторов, с которым может работать модель. Сегодня существует большое число моделей, учитывающих куда большее число параметров, чем игра в «жизнь» (например, такие клеточные автоматы, как игра «Пожар» и подобные). Очевидно, что чем больше параметров учитывает система, тем она точнее предсказывает поведение социума).

2. Разработать систему параметров, определяющих факторы, которая может стабильно работать с большинством социальных ситуаций.

Таким образом, определено направление дальнейшего изучения симуляций социального поведения. Исследованы многие способы симуляции общества. Определены цели, которых необходимо достигнуть для получения как можно более точной модели симуляции социального поведения.

Литература

1. *Теннис, Ф.* Общность и общество // Социологический журнал. – 1998.– № 3–4. – С. 226.

2. СМИ «МЕДУЗА». – URL: https://meduza.io/feature/2020/03/30/v-moskve-vveli-zhestkie-karantinnye-mery-pohozhe-eto-pravilno-matematicheskaya-model-pokazyvaet-chto-inache-mogli-by-pogibnut-bolshe-100-tysyach-chelovek

3. Particle Swarm Optimization // Proceedings of IEEE International Conference on Neural Networks IV. – 1995. – P. 1942–1948.

4. Kennedy, J.; Eberhart, R.C. Swarm Intelligence (неопр.). – Morgan Kaufmann, 2001.

Шенгелия Артем Геннадьевич – студент 1-го курса магистратуры кафедры ERP-систем и бизнес процессов Воронежского государтсвенного университета. E-mail: Artem-shengeliya@yandex.ru

Сафронов Виталий Владимирович (научный руководитель) – канд. техн. наук, доц., доцент кафедры кафедры ERP-систем и бизнес процессов Воронежского государственного университета. E-mail: vitolik@bk.ru

УДК 004.42

ПРИЛОЖЕНИЕ ДЛЯ РАСЧЁТА ЭКОНОМИЧЕСКИХ РИСКОВ НА ОСНОВЕ СИМПЛЕКСНОГО МЕТОДА

О. Ю. Шишкина

Воронежский государственный университет

Введение

В статье описывается методика, позволяющая спрогнозировать прибыль предприятия на основе данных о производстве для различных диапазонов доли гарантированных продаж, и архитектура приложения, разработанного на её основе.

Основой предложенной методики является поиск решения задачи линейного программирования, а именно – симплексный метод.

Результат проведенного программой анализа представлен в виде графика, показывающего, как изменяется чистая финансовая прибыль при росте доли гарантированных продаж. Каждая целая (по оси абсцисс) точка графика имеет полное описание того, какие действия необходимо предпринять, чтобы достичь соответствующей прибыли.

1. Функциональность приложения

1.1. Анализ функциональности приложения

Функциональность приложения можно условно разделить на две части:

1. Ввод данных о производстве.
2. Анализ данных.

Подробнее о требованиях к функциональности можно узнать, рассмотрев диаграмму прецедентов (рис. 1).

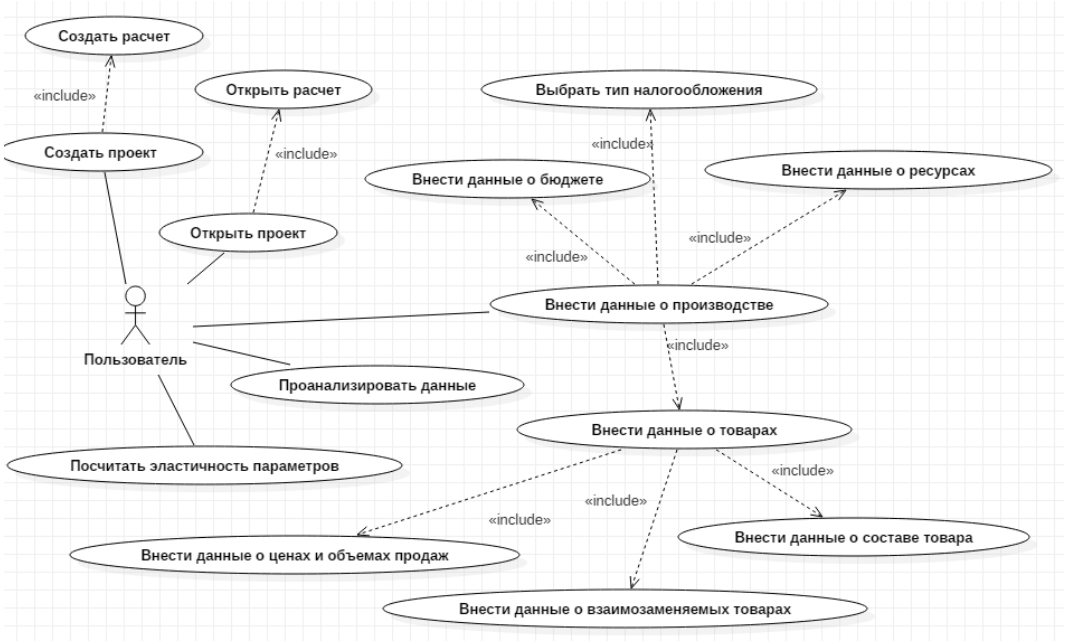


Рис. 1. Диаграмма прецедентов

1.2. Необходимые данные о производстве

Для того чтобы оптимально распределить ресурсы производства и оценить рискованность различных комбинаций цен и объемов, пользователь должен внести следующие данные:

1. Бюджет предприятия.
2. Система налогообложения.
3. Данные о постоянных доходах и расходах.
4. Данные о ресурсах производства (пример на рис. 2):
   а. Название.
   б. Стоимость удельной единицы ресурса.
   в. Доступный максимум.
   г. Запасы на складе.
   д. Следует ли исчислять ресурс как целочисленный (да или нет).
   е. Является ли ресурс скоропортящимся (да или нет).

Рис. 2. Пример данных о ресурсах

5. Данные о товарах или услугах (рис. 3):
   а. Название.
   б. Необходимый минимум производства.
   в. Запасы на складе.
   г. Следует ли исчислять товар (услугу) как целочисленный (да или нет).
   д. Является ли товар (услуга) скоропортящимся (да или нет).

Рис. 3. Пример данных о товарах и услугах

е. Состав товара (услуги) – список ресурсов и их количество, необходимых для производства одной единицы наименования (рис. 4).

ж. Информация о ценовых комбинациях (рис. 5).

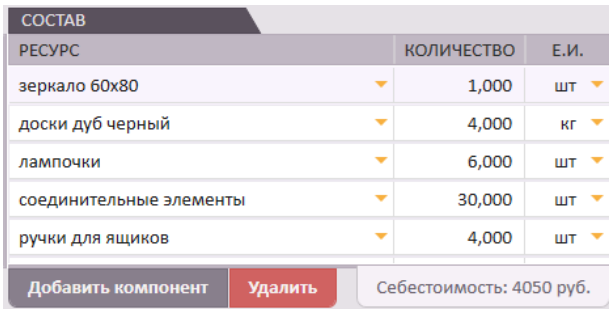


Рис. 4. Пример состава товара или услуги

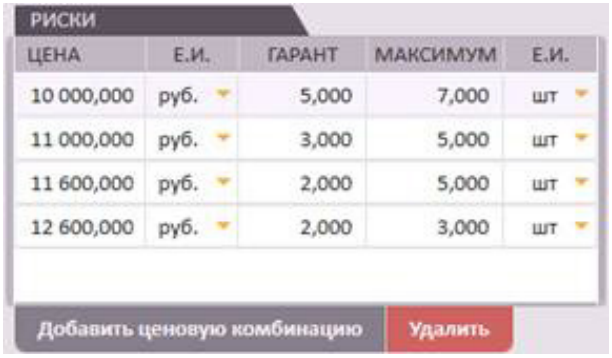


Рис. 5. Пример списка ценовых комбинаций товара или услуги

Информация о ценовых комбинациях – это разнообразие цен и объемов необходимое для того, чтобы перебрать все возможные комбинации цен товаров и выбрать лучшую для указанной доли гарантированных продаж.

Для того чтобы составить график различных вариантов рисков, пользователю предлагается описать цены и объемы продаж товаров и услуг следующим образом: необходимо указать несколько цен, по которым может быть реализовано выбранное наименование. Каждому варианту цены соответствует интервал продаж – от гарантированного до максимально возможного.

1.3. Хранение данных

Информация, введенная пользователем, хранится в базе данных. В качестве СУБД выбрана SQLite по ряду причин:

1. Конфиденциальность данных – файл базы данных хранится локально на компьютере пользователя.

2. Простота хранения структуры проектов – структура представляет собой вложенные каталоги и файлы базы данных.

2. Анализ данных

2.1. Преобразование данных в задачу линейного программирования

После того, как данные о производстве получены, следующим этапом является их преобразование в математическую модель.

Описанный далее алгоритм создания математической модели зависит от доли гарантированных продаж α, где $0 < \alpha \leq 1$. Пункты 1–3 являются общими для всех долей и вычисляются один раз, а следующие этапы зависят от выбранной доли гарантированных продаж (рисков).

Преобразование данных делится на следующие этапы:

1. Проверка наличия противоречий в данных.

Например, минимальное необходимое количество товара использует больше ресурса, чем его возможно закупить.

2. Составление всевозможных комбинаций цен товаров и услуг.

Определение. Пусть существует n наименований товаров и услуг, каждое i-е наименование имеет p_i различных ценовых комбинаций. Тогда общее количество всевозможных комбинаций R вычисляется по формуле:

$$R = \prod_{i=1}^{n} p_i. \qquad (1)$$

Предположим, что имеется 15 товаров, каждый из которых описывается 3-мя ценовыми комбинациями, тогда общее количество комбинаций R равно $3^{15} = 14348907$. Заметим, что получилось достаточно большое число при небольшом количестве входных данных.

3. Составление общей части системы линейных неравенств, основывающейся на данных о составе товаров, ограничениях ресурсов, ограничении бюджета, минимумах товаров или услуг.

Определение.

i. Пусть имеется n наименований товаров и услуг и m наименований ресурсов. Пусть вектор $B = (b_1, b_2, ..., b_m)^\tau$ описывает максимумы ресурсов r_1, r_2,... r_m. А матрица

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \dots & \dots & \dots & \dots \\ a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{pmatrix} \tag{2}$$

описывает потребление ресурсов в производстве товаров или услуг, где $a_{i,j}$ – количество ресурса r_i в составе товара t_j. Пусть вектор $X = (x_1, x_2, ..., x_n)$ характеризует количество производимых единиц каждого наименования товаров или услуг.

Тогда справедливо сравнение:

$$A \times X \le B. \tag{3}$$

ii. Кроме того, нужно не забывать про бюджет. Пусть с учетом дополнительных доходов и расходов, денежный поток ограничивается числом φ (вычисляется как бюджет плюс дополнительные доходы минус дополнительные расходы). Введем вектор $C = (c_1, c_2, ..., c_n)$, содержащий себестоимости одной единицы каждого товара или услуги. Тогда имеет место ограничение на бюджет:

$$C \times X \le \varphi. \tag{4}$$

iii. Чтобы учесть запасы товаров, необходим вектор $X' = (x'_1, x'_2, ..., x'_n)^\tau$, где x'_i – количество единиц i-го товара, взятого из запаса. Так как запас ограничен вектором $St = (st_1, st_2, ..., st_n)^\tau$, где st_i – запасы i-го товара, то справедливо неравенство:

$$\theta \le X' \le St. \tag{5}$$

iv. Также каждый из товаров имеет минимум, вектор $L = (l_1, l_2, ..., l_n)^\tau$ описывает ограничения снизу на каждый товар или услугу. Из этого получаем неравенство:

$$L \le X + X'. \tag{6}$$

v. Из ограничений (3)–(6) следует система неравенств:

$$\begin{cases} A \times X \le B \\ C \times X \le \varphi \\ \theta \le X' \le St \\ L \le X + X' \end{cases}. \tag{7}$$

Данная система является общей частью для итоговой системы каждой комбинации.

4. Для каждой из R комбинаций необходимо составить дополнительные неравенства и целевую функцию.

Определение.

Рассмотрим некоторую комбинацию цен товаров или услуг. Пусть вектор $S = (s_1, s_2, ..., s_n)$, где $1 \le s_i \le p_i$, описывает для каждого i-го наименования товара или услуги, какая ценовая комбинация была выбрана, здесь s_i – индекс ценовой комбинации. Каждая такая ценовая комбинация представляет собой три числа: цену pr_i, гарант g_i и максимум M_i. В зависимости

от выбранной доли гарантированных продаж α ограничение сверху на объем производства товара вычисляется по формуле:

$$q_{\alpha,i} = \min\left(\frac{g_i}{\alpha}, M_i\right). \quad (8)$$

Тогда вектор $Q_\alpha = (q_{\alpha,1}, q_{\alpha,2}, ..., q_{\alpha,n})^\tau$ ограничивает сверху объемы продаж каждого товара. Таким образом, справедливо неравенство:

$$X + X' \le Q_\alpha. \quad (9)$$

Это неравенство добавляется к системе (6).

Составим целевую функцию. Пусть вектор $P = (pr_1, pr_2, ..., pr_n)$ описывает выбранные цены товаров или услуг. Тогда функция

$$F = P \times (X - C) + P \times X' \quad (10)$$

является целевой функцией прибыли. А векторы X и X' – искомые решения.

2.2. Решение задачи линейного программирования при условии целочисленности переменных

Итак, результатом алгоритма, представленного в пункте 2.1, для каждой комбинации цен товаров и услуг является система линейных неравенств и целевая функция. Такая задача, т. е. задача максимизации целевой функции при условии выполнения соответствующих неравенств, имеет общеизвестный алгоритм решения симплексным методом. Следует отметить, что симплексный метод дает решения с учетом того, что элементы искомых векторов X и X' могут принимать любые значения (не обязательно целые).

Таким образом, предложение произвести полтора компьютера будет невыполнимым, хотя и очень выгодным. Для решения задачи частичного целочисленного линейного программирования был выбран метод Лэнд и Дойг (метод ветвления и границ).

2.3. Выбор лучших комбинаций и составление графика

Для составления графика пользователь выбирает интервал долей гарантированных продаж, $0 < \alpha_l < \alpha_r \le 1$ (α кратно 0,01). Шаг – 1 %, то есть 0,01. Общее количество точек графика вычисляется по формуле:

$$N_{graph} = \frac{\alpha_r - \alpha_l}{0,01} + 1. \quad (11)$$

Для каждой доли гарантированных продаж $\alpha_i = \alpha_l + i \cdot 0,01$, где $0 \le i \le N_{graph} - 1$, составляется R (формула 1) задач целочисленного линейного программирования. Далее для каждой задачи необходимо найти решение. Та задача и ее решение, которые дают максимальное значение целевой функции F относительно других задач, выбираются для описания данной точки с абсциссой $\alpha_i \cdot 100$ (целое значение процентов). А само значение целевой функции F – ее ординатой.

Помимо основного графика чистой прибыли (зеленый график на рис. 6), также могут быть построены различные дополнительные графики, опирающиеся на уже имеющееся решение. Например, график гарантированных продаж, характеризующий прибыль при условии того, что продажа товаров и услуг сверх гарантированного объема не оказалась успешной (красный график на рис. 6).

Стоит заметить, что полученный график чистой прибыли в случае успеха, построенный по описанному алгоритму, является невозрастающим. Доказательство этого приведено ниже.

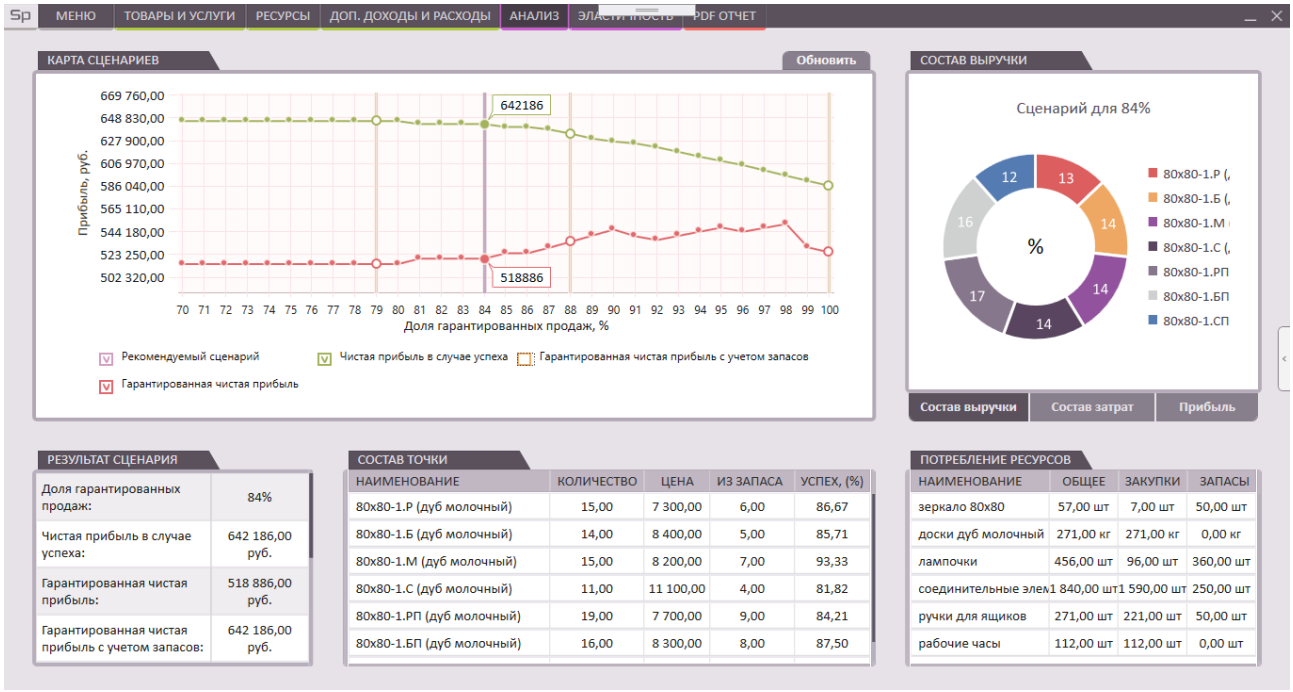


Рис. 6. Пример результата работы программы

Пусть имеются две точки (α_i, F_i) и (α_j, F_j) (то есть рассмотрим случай, когда решения в этих точках существуют, если не существует, то и точек нет на графике), необходимо доказать, что при $i < j$: $F_i \geq F_j$.

Так как от значения α зависит только верхняя граница $X'' = X + X'$, из п.8,9, то $Q_{\alpha_i} \geq Q_{\alpha_j}$. Для любой из R комбинаций цен товаров и услуг, диапазон X'' при $\alpha = \alpha_j$ равен $L \leq X''_j \leq Q_{\alpha_j}$, а при $\alpha = \alpha_i$: $L \leq X''_i \leq Q_{\alpha_i}$, так как $Q_{\alpha_i} \geq Q_{\alpha_j}$, то $L \leq X''_j \leq Q_{\alpha_j} \subseteq L \leq X''_i \leq Q_{\alpha_i}$. Так как среди X''_j имеется решение задачи для $\alpha = \alpha_j$, оно же удовлетворяет условию задачи для $\alpha = \alpha_i$, при этом остальные условия системы линейных неравенств одинаковые, следовательно, для задачи $\alpha = \alpha_i$ решение может принести такое же значение целевой функции, как и для $\alpha = \alpha_j$ или даже больше, так как интервал возможных X''_i такой же или шире. Следовательно $F_i \geq F_j$. Что и требовалось доказать.

3. Проблема больших данных

Как уже было замечено, число R – количество комбинаций для каждой доли гарантированных продаж, а значит и количество задач целочисленного линейного программирования, быстро растет с увеличением объема входных данных. Общее количество задач целочисленного линейного программирования оценивается, как:

$$R \cdot N_{graph} = \prod_{i=1}^{n} p_i \cdot \left(\frac{\alpha_r - \alpha_l}{0,01} + 1 \right). \quad (12)$$

В свою очередь, каждому алгоритму ветвления и границ требуется количество запусков решений задачи обычного линейного программирования, экспоненциально зависящее от количества товаров.

В реализованной на данный момент программе стоят некоторые ограничения, например, количество точек графика не более 30, количество товаров не более 10, а количество вариантов

ценовых комбинаций для каждого товара не более 5. Таким образом, согласно формуле (12), получается число задач целочисленного линейного программирования:

$$5^{10} \cdot 30 = 292{,}968{,}750.$$

Для решения подобной задачи предлагается использовать параллельные вычисления (количество потоков вдвое больше количества ядер компьютера). Так как решения разных задач целочисленного линейного программирования никак не зависят друг от друга, эти процессы можно легко распределить на разные параллельные потоки.

Кроме того, разработаны дополнительные методы оптимизации, уменьшение количества вычислений за счет отсева ценовых комбинаций товара или услуги, которые заведомо не могут быть выбраны алгоритмом в качестве лучших. Но, тем не менее, в самом худшем из возможных случаев, программа столкнется с ситуацией, когда нужно будет посчитать все 292,968,750 задач целочисленного линейного программирования. Это может занять продолжительное время: десятки минут или даже часов.

На данный момент, для использования программы локально на одном компьютере, реализовано следующее решение: описанные выше ограничения накладываются на один «расчет», и таких расчетов можно создавать достаточно много и объединять результаты их вычислений в один график.

Например, пусть имеется 100 товаров (остальные ограничения прежние), если все 100 товаров поместить в один расчет, получится $5^{100} \cdot 30$, если же распределить 100 товаров на 10 расчетов по 10 товаров в каждом, то получится $5^{10} \cdot 30 \cdot 10$. Разница колоссальна, но точность вычислений при этом падает, поэтому разделять товары на расчеты нужно с условием, что расчеты друг от друга не зависят.

С сегодняшними технологиями и возможностями облачных вычислений, данную программу можно улучшить за счет использования внешних ресурсов, эффективно распределив нагрузку, и тогда ограничения на входные данные не будут такими строгими.

Заключение

Статья описывает архитектуру и математическую основу приложения для оптимального распределения ресурсов производства и учета экономических рисков. Реализованная программа позволяет применить на практике известную задачу линейного программирования. Ее основные принципы были описаны Л. В. Канторовичем еще в 1939-м году, но не нашли общего применения в советской плановой экономике.

На сегодняшний день в большинстве предприятий этап оптимального планирования и распределения ресурсов пропускается или выполняется на основе приблизительных оценок экономистов. Приложение, описанное в данной статье, позволит выполнять этот этап качественно, и получать математически выверенные прогнозы, учитывая при этом опыт специалистов и статистические данные.

Литература

1. *Канторович, Л. В.* Математико-экономические труды / Л. В. Канторович. – Новосибирск : Наука, 2011. – 760 с.

2. *Корбут, А. А.* Дискретное программирование / А. А. Корбут, Ю. Ю. Финкельштейн. – Москва : Наука, 1969. – 366 с.

3. *Азарнова, Т. В.* Методы оптимизации. Элементы теории, алгоритмы и примеры / Т. В. Азарнова, И. Л. Каширина, Г. Д. Чернышова. – Воронеж : Воронеж. гос. ун-т, 2004. – 151 с.

4. *Аснина, А. Я.* Оптимизационные задачи в экономике : практикум / А. Я. Аснина Н. Г. Аснина, О. С. Нильга ; Воронеж гос. арх.-строит. ун-т. – Воронеж, 2009. – 68 с.
5. Ашманов, С. А. Линейное программирование / С. А. Ашманов. – М. : Наука, 1981. – 304 с.

Шишкина Ольга Юрьевна – студент 4-го курса кафедры ПО и АИС Воронежского государственного университета. E-mail: govorkova.t@yandex.ru

Старикова Анна Александровна (научный руководитель) – канд. техн. наук, преподаватель кафедры ПО и АИС Воронежского государственного университета. E-mail: astar.box@gmail.com

УДК 004.9

ПРЕИМУЩЕСТВА И НЕДОСТАТКИ ПРЕФИКСНОГО СЖАТИЯ ИНДЕКСОВ

Т. С. Щеглова

Воронежский государственный университет

Введение

В основном, все информационные системы в настоящее время используют данные, которые организованы специальным образом с помощью систем управления базами данных (СУБД). Однако, таблицы в таких системах могут иметь большое количество строк, хранящихся в произвольном порядке, и их поиск по какому-либо заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. В связи с этим, для того чтобы обеспечить наиболее быстрый поиск, были введены индексы – наборы дополнительных данных, записанные в удобном виде, которые позволяют существенно ускорить поиск, хоть и требуют дополнительных усилий для поддержания их актуальности [1].

Индекс состоит из значений одного или нескольких столбцов таблицы и указателей на соответствующие строки таблицы и, таким образом, позволяет искать строки, удовлетворяющие критерию поиска. В зависимости от типов баз данных, индексы могут быть различными и реализовываться за счёт специфических математических механизмов. Наиболее часто индекс имеет оптимизированную под поиск структуру сбалансированного дерева, так как поддерживать такой индекс относительно просто.

Однако постоянно возрастающие требования к индексам как по скорости их работы, так и по минимизации объема хранимой информации, делают всегда актуальным рассмотрение новых вариантов. В данной работе будет рассмотрено использование сжатых префиксных деревьев для организации индексов.

1. Префиксное дерево

Для начала рассмотрим, что такое префиксное дерево, которое также называют нагруженным деревом. Это структура данных, реализующая интерфейс ассоциативного массива, то есть позволяющая хранить пары «ключ-значение» [2]. Нагруженное дерево отличается от обычных деревьев поиска тем, что в его узлах не хранятся ключи. Вместо них в узлах хранятся односимвольные метки, а ключом, который соответствует некоторому узлу, является путь от корня дерева до этого узла, а точнее строка, составленная из меток узлов, повстречавшихся на этом пути. В таком случае корень дерева соответствует пустому ключу.

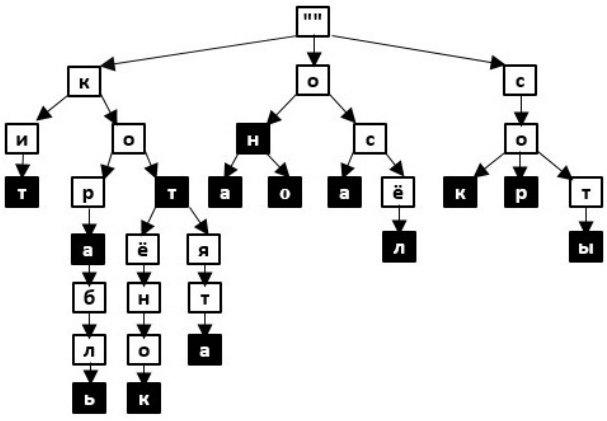


Рис. 1. Пример префиксного дерева

На рис. 1 некоторые узлы префиксного дерева выделены чёрным, это означает, что в них может содержаться последний символ ключа, остальные узлы называются промежуточными. Считается, что префиксное дерево содержит строку, соответствующую ключу, только в том случае, когда эту строку можно прочитать, дви-

гаясь из корня до некоторого выделенного узла. На рис. 1 ключами являются строки: кит, кора, корабль, кот, котёнок, котята, он, она, оно, оса, осёл, сок, сор, соты.

2. Сжатое префиксное дерево

Для достижения более оптимального варианта к префиксному дереву применяют процедуру сжатия. Сжатое нагруженное дерево образуется из классического дерева путём удаления промежуточных узлов, ведущих к одному узлу, который не является промежуточным [3]. На рис. 2 приведён пример префиксного сжатого дерева.

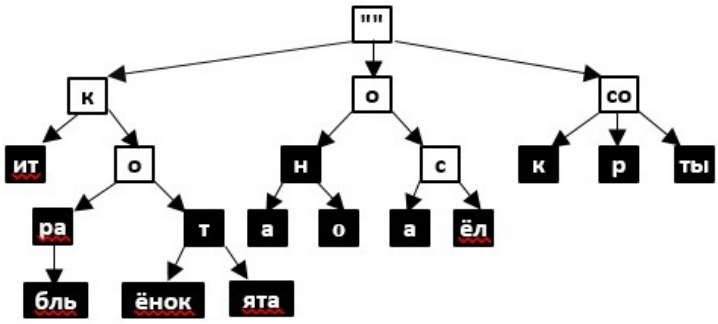


Рис. 2. Пример сжатого префиксного дерева

В программной реализации для определения, является ли данный узел выделенным, то есть можно ли, остановившись на нём, построить ключ, к каждому не промежуточному узлу добавляется терминальный символ, который нигде в строке больше не встречается, например, символ конца строки. И далее все цепочки строк, которые заканчиваются терминальным символом, будут рассматриваться в качестве ключей.

Для программной реализации будет храниться в каждом узле список дочерних узлов. Список будет односвязным и храниться в узле будет только ссылка на голову списка, так называемую, старшую дочь. Это даёт преимущество в том, что можно отказаться от пустого корня. Таким образом, не листовые узлы могут содержать два указателя: link – на старший дочерний узел, next – на свою младшую сестру. На рис. 3 показан процесс такого преобразования, сплошные стрелки соответствуют указателям link, пунктирные – указателям next.

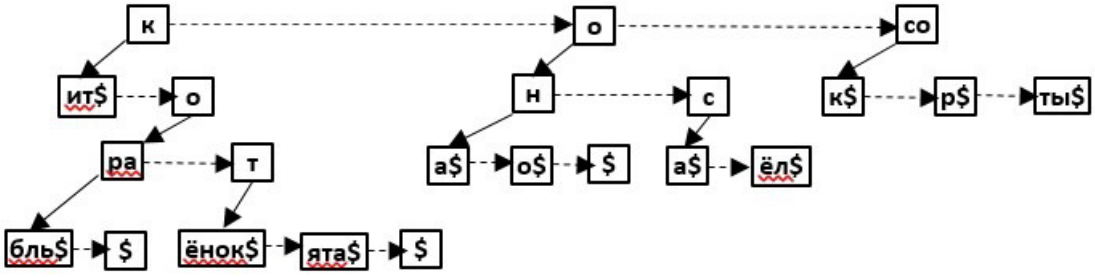


Рис. 3. Представление сжатого префиксного дерева в программной реализации

3. Операции поиска, добавления и удаления для сжатого префиксного дерева

Рассмотрим для начала операцию поиска нужного ключа по префиксному дереву. Движение начинаем от корневого узла дерева, на рис. 3 такой узел содержит строку «к» и является первым элементом верхнего списка. Если данный узел пустой, то поиск заканчивается с отрицательным результатом. Иначе, происходит сравнение искомой строки x, считается, что она заканчивается символом конца строки, и значения в узле str и вычисляется длина их наибольшего общего префикса. Далее могут быть получены следующие случаи:

– общий префикс может быть пустым, тогда надо рекурсивно продолжать поиск в последующей сестре текущего узла, то есть перейти по ссылке next;

– общий префикс может быть равен искомой строке x, и в этом случае, если узел листовой, то поиск заканчивается успешно;

– общий префикс может совпадать с ключом, но не совпадать с x, в данной ситуации нужно рекурсивно перейти по ссылке link к старшему дочернему узлу и передать ему для поиска строку x без найденного ранее префикса;

– общий префикс может быть, но не совпадать со строкой в узле, тогда поиск считается неудачным.

Теперь рассмотрим вставку нового ключа в сжатое префиксное дерево. Данная операция основана на поиске ключа и далее уже добавлении нового узла при отрицательном результате поиска. Рассмотрим нижеуказанные варианты:

– в случае пустого дерева нужно создать новый узел с заданным ключом;

– если список дочерних объектов пройден и не найдено общего префикса строки текущего узла str и добавляемой строки x, то у последнего элемента списка сделать ссылку next на вновь созданный узел, содержащий значение x;

– в ситуации, когда длина общего префикса строки текущего узла str и добавляемой строки x больше нуля, но меньше длины str, нужно разбить текущий узел на два, оставив в родительском узле найденный префикс, а из него сделать ссылку link на дочерний узел p, поместив в него оставшуюся часть строки родительского узла. После разделения процесс вставки не заканчивается, нужно добавить к узлу p ссылку next на вновь созданный узел, содержащий строку x без найденного префикса.

И в завершении проанализируем операцию удаления из сжатого префиксного дерева. При удалении ключа удаляется всего один листовой узел, соответствующий суффиксу некоторого удаляемого ключа. Сначала с помощью операции поиска необходимо найти этот узел, если поиск успешный, то будут предприняты действия по удалению. Если листовой узел был найден по ссылке next или link, то нужно перевесить соответствующую ссылку на последующую сестру, если она имеется, а найденный узел удалить. В случае, если удалённый узел был найден по ссылке link, и у него больше нет младших сестёр, необходимо произвести слияние узлов для поддержания префиксного дерева сжатым. Для этого значение из листового узла, который впоследствии удаляется, добавляется к родительскому узлу, который сам теперь становится листовым.

4. Сравнительный анализ

Для того чтобы получить наглядное представление о рассмотренных структурах данных, было проведено численное исследование по сравнению таких структур данных, как B-дерево, префиксное и сжатое префиксное дерево по параметрам быстродействия и занимаемого объема памяти. В качестве набора данных для тестирования использовались случайные числа из диапазона от 100 до 1000000 в количестве 100000. На рис. 4 представлен сравнительный анализ быстродействия по таким показателям, как построение дерево и поиск всех значений. По данной диаграмме можно понять, что построение дерева не на много, но всё же быстрее у B-деревьев, а вот по поиску ключей B-деревья имеют уже значительное преимущество над префиксными и сжатыми префиксными деревьями. На рис. 5 изображён сравнительный анализ размерности структур в байтах. И отсюда можно сделать вывод, что сжатые префиксные деревья занимают значительно меньше памяти по сравнению с другими представленными структурами данных.

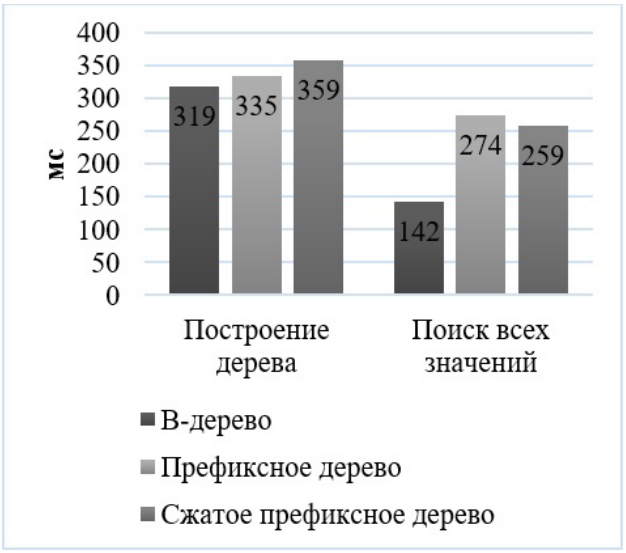


Рис. 4. Сравнительный анализ быстродействия

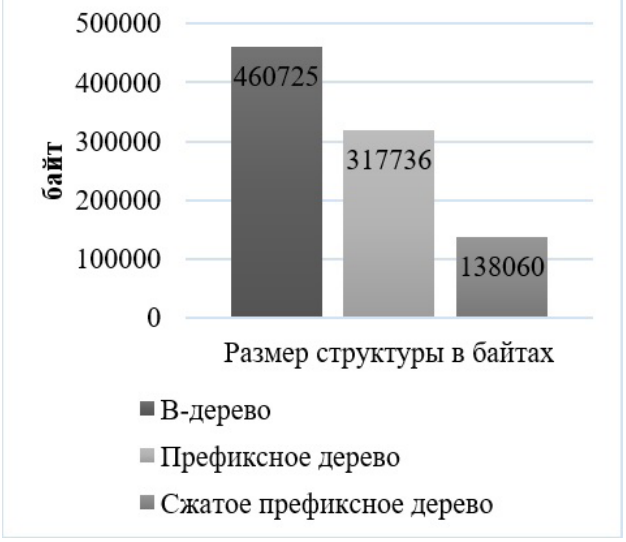


Рис. 5. Сравнительный анализ размерности

Заключение

В данной статье были рассмотрены такие структуры данных, как префиксные и сжатые префиксные деревья, которые могут подойти для организации работы индексов. Были проанализированы операции поиска, вставки и удаления для сжатых нагруженных деревьев. К сожалению, поиск по такой структуре данных не всегда будет происходить быстрее, чем у сбалансированных деревьев, так как в худшем случаем требуется перебор по списку всех сестёр дочернего объекта на каждом уровне дерева. Однако данная структура занимает значительно меньше памяти за счёт сжатия и может использоваться в качестве индексов при требовании экономии памяти.

Литература

1. Википедия [Электронный ресурс]: Database_index. – Режим доступа: https://en.wikipedia.org/wiki/Database_index. – (Дата обращения: 15.03.2019).

2. *Ахо, А. В.* Структуры данных и алгоритмы / А. В. Ахо, Дж. Э. Хопкрофт, Дж. Д. Ульман, под ред. С. Н. Тригуба, пер. с англ. А. А. Минько. – Москва : Вильямс, 2003. – 384 с.

3. *Ершов Николай* [Электронный ресурс]: сжатые префиксные деревья. – Режим доступа: https://habr.com/ru/post/151421/. – (Дата обращения: 27.10.2019).

Щеглова Татьяна Сергеевна – студент 2-го курса магистратуры кафедры МОЭВМ Воронежского государственного университета. E-mail: tanufa25@mail.ru

Борисенков Дмитрий Васильевич (научный руководитель) – канд. техн. наук, доцент кафедры МОЭВМ Воронежского государственного университета. E-mail: xuser@relex.ru

Содержание

Научное издание

Математика,
информационные технологии,
приложения

Сборник трудов
Межвузовской научной конференции
молодых ученых и студентов

Воронеж,
27 апреля 2020 г.

Подписано в печать 27.05.2020. Формат 60×84/8.
Усл. печ. л. 38,4. Тираж 100 экз. Заказ № 89.

Отпечатано с готового оригинал-макета

ООО Издательско-полиграфический центр
«Научная книга»
394030, г. Воронеж, ул. Никитинская, д. 38, оф. 308
Тел. +7 (473) 200-81-02, 200-81-04
http://www.n-kniga.ru; e-mail: zakaz@n-kniga.ru

Отпечатано в типографии ООО ИПЦ «Научная книга».
394026, г. Воронеж, Московский пр-т, 11б
Тел. +7 (473) 220-57-15, 238-02-38
http://www.n-kniga.ru; e-mail: typ@n-kniga.ru