МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
МЕЖДУНАРОДНАЯ АКАДЕМИЯ ИНФОРМАТИЗАЦИИ
СОЮЗ МАШИНОСТРОИТЕЛЕЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
МИНИСТЕРСТВО ПРОМЫШЛЕННОСТИ, ИННОВАЦИОННЫХ
И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ РЯЗАНСКОЙ ОБЛАСТИ
РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ В.Ф. УТКИНА

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ

СТНО-2020

III МЕЖДУНАРОДНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ФОРУМ

Сборник трудов

Том 4

Рязань
Book Jet
2020

УДК 004 + 001.1 + 681.2+ 681.2+ 681.3+681.5
С 568

Современные технологии в науке и образовании – СТНО-2020 [текст]: сб. тр. III междунар. науч.-техн. форума: в 10 т. Т.4./ под общ. ред. О.В. Миловзорова. – Рязань: Рязан. гос. радиотехн. ун-т, 2020; Рязань. – 198 с.,: ил.

Сборник включает труды участников III Международного научно-технического форума «Современные технологии в науке и образовании» СТНО-2020.

В сборнике освещаются вопросы математического моделирования, новых технологий в радиотехнике, телекоммуникациях, электротехнике и радиоэлектронике, вопросы полупроводниковой наноэлектроники, приборостроения, лазерной, микроволновой техники, силовой промышленной электроники, новые технологии в измерительной технике и системах, биомедицинских системах, алгоритмическое и программное обеспечение вычислительной техники, вычислительных сетей и комплексов, вопросы систем автоматизированного проектирования, обработки изображений и управления в технических системах, перспективные технологии в машиностроительном и нефтехимическом производствах, новые технологии и методики в высшем образовании, в т.ч. вопросы гуманитарной и физико-математической подготовки студентов, обучения их иностранным языкам, перспективные технологии электронного обучения, в том числе, дистанционного, вопросы экономики, управления предприятими и персоналом, менеджмента, а также вопросы гуманитарной сферы.

Авторская позиция и стилистические особенности сохранены.

УДК 004 + 001.1 + 681.2+ 681.2+ 681.3+681.5

ISBN 978-5-7722-0301-9

ИНФОРМАЦИЯ О III МЕЖДУНАРОДНОМ ФОРУМЕ «СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ» СТНО-2020

III Международный научно-технический форум «Современные технологии в науке и образовании» СТНО-2020 состоялся 04.03.2020-06.03.2020 в г. Рязань в Рязанском государственном радиотехническом университете имени В.Ф. Уткина.

В рамках форума «Современные технологии в науке и образовании» СТНО-2020 состоялась работа четырех Международных научно-технических конференций:

«Современные технологии в науке и образовании. Радиотехника и электроника», секции

- Радиотехнические системы и устройства;
- Телекоммуникационные системы и устройства;
- Цифровые информационные технологии реального времени;
- Промышленная силовая электроника, электроэнергетика и электроснабжение;
- Физика полупроводников, микро- и наноэлектроника;
- Микроволновая, оптическая и квантовая электроника;
- Актуальные задачи химических технологий;

«Современные технологии в науке и образовании. Вычислительная техника и автоматизированные системы», секции

- Алгоритмическое и программное обеспечение вычислительных систем и сетей;
- ЭВМ и системы;
- Системы автоматизированного проектирования;
- Информационные системы и защита информации;
- Математические методы в научных исследованиях;
- Обработка изображений и управление в технических системах;
- Геоинформационные и космические технологии;
- Автоматизация производственно-технологических процессов в приборо- и машиностроении;
- Информационно-измерительные устройства и системы в технике и медицине. Стандартизация и управление качеством;
- Информационные системы и технологии;

«Современные технологии в науке и образовании. Экономика и управление», секции;

- Современные технологии государственного и муниципального управления;
- Экономика, менеджмент и организация производства;
- Бухгалтерский учет, анализ и аудит;
- Управление персоналом;
- Экономическая безопасность;

«Современные технологии в науке и образовании. Новые технологии и методы в высшем образовании», секции

- Современные технологии электронного обучения;
- Иностранный язык в техническом вузе;
- Лингвистика и межкультурная коммуникация;
- Направления и формы гуманитаризации высшего образования и гуманитарная подготовка студентов;
- Методы преподавания и организация учебного процесса в вузе;
- Физико-математическая подготовка студентов;
- Технологии обучения и воспитания на военной кафедре.

Организационный комитет Форума:

Чиркин М.В., ректор, д.ф.-м.н., проф. – председатель

Гусев С.И., проректор по научной работе, д.т.н., проф. – зам. председателя;

Бухенский К.В., зав. кафедрой высшей математики, к.ф.-м.н., доц. – зам. председателя;

Миловзоров О.В., зам. директора института магистратуры и аспирантуры, к.т.н, доц. – координатор;

Устинова Л.С., начальник отдела информационного обеспечения – отв. за информационную поддержку;

Трубицына С.Г., вед. инженер – секретарь оргкомитета;

Благодарова И.А., ведущий программист – секретарь оргкомитета;

члены оргкомитета:

Авилкина С.В., доцент кафедры государственного, муниципального и корпоративного управления, к.п.н., доц.;

Алпатов Б.А., профессор кафедры автоматики и информационных технологий в управлении, д.т.н., проф.;

Бабаян П.В., проректор по учебной работе, зав. кафедрой автоматики и информационных технологий в управлении, к.т.н., доц.;

Витязев В.В., зав. кафедрой телекоммуникаций и основ радиотехники, д.т.н., проф.;

Евдокимова Е.Н., зав. кафедрой экономики, менеджмента и организации производства, д.э.н., проф.;

Еремеев В.В., директор НИИ «Фотон», д.т.н., проф.;

Есенина Н.Е., зав. кафедрой иностранных языков, к.п.н., доц.;

Животягин Д.А. нач. кафедры связи военного учебного центра, полковник;

Жулев В.И., зав. кафедрой информационно-измерительной и биомедицинской техники, д.т.н., проф.;

Кириллов С.Н., зав. кафедрой радиоуправления и связи, д.т.н., проф.;

Клейносова Н.П., директор центра дистанционного обучения, к.п.н., доц.;

Клочко В.К., профессор кафедры автоматики и информационных технологий в управлении, д.т.н., проф.;

Коваленко В.В., зав. кафедрой химической технологии, к.т.н., доц.;

Корячко В.П., д.т.н., проф., зав. кафедрой систем автоматизированного проектирования вычислительных средств;

Костров Б.В., зав. кафедрой электронных вычислительных машин, д.т.н., проф.;

Кошелев В.И., зав. кафедрой радиотехнических систем, д.т.н., проф.;

Куприна О.Г., доцент кафедры иностранных языков, к.филол.н., доц.;

Круглов С.А., зав. кафедрой промышленной электроники, к.т.н., доц.;

Лукьянова Г.С., доцент кафедры высшей математики, к.ф.-м.н., доц.;

Мусолин А.К., зав. кафедрой автоматизации информационных и технологических процессов, д.т.н., проф.;

Овечкин Г.В., зав. кафедрой вычислительной и прикладной математики, д.т.н., проф.;

Паршин Ю.Н., зав. кафедрой радиотехнических устройств, д.т.н., проф.;

Перфильев С.В., зав. кафедрой государственного, муниципального и корпоративного управления, д.э.н., проф.;

Пржегорлинский В.Н., зав. кафедрой информационной безопасности, к.т.н., доц.;

Пылькин А.Н., профессор кафедры вычислительной и прикладной математики, д.т.н., проф.;

Рохлина Т.А., доцент кафедры иностранных языков, к.филол.н., доц.;

Серебряков А.Е., зам. зав. кафедрой электронных приборов, к.т.н.;

Соколов А.С., зав. кафедрой истории, философии и права, д.и.н;

Таганов А.И., зав. кафедрой космических технологий, д.т.н., проф.;

Токарь А.Д., нач. кафедры ВКС военного учебного центра, полковник;

Холомина Т.А., зав. кафедрой микро- и наноэлектроники, д.ф.-м.н., проф.;

Холопов С.И., декан ф-та автоматики и информационных технологий в управлении, зав. кафедрой автоматизированных систем управления, к.т.н., доц.;

Чеглакова С.Г., зав. кафедрой экономической безопасности, анализа и учета, д.э.н., проф..

МЕЖДУНАРОДНАЯ НАУЧНО-ТЕХНИЧЕСКАЯ КОНФЕРЕНЦИЯ «СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ. ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ»

СЕКЦИЯ «АЛГОРИТМИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СЕТЕЙ»

УДК 519.16; ГРНТИ 27.45.15

ОПЕРАЦИИ С СОЧЕТАНИЯМИ В ПРОГРАММИРОВАНИИ

А.В. Пруцков

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Россия, Рязань, http://prutzkow.com

Аннотация. Комбинаторика широко используется в информатике и в программировании в частности. Операции с сочетаниями, такие как их нумерация, генерация и кластеризация, используются при программировании некоторых задач, тестировании программ и анализе алгоритмов. Приводятся наиболее эффективные способы выполнения операций с сочетаниями для дальнейшего использования в научных и учебных целях.

Ключевые слова: комбинаторика, сочетания, подмножества, нумерация сочетаний, получение всех сочетаний.

COMBINATION OPERATIONS IN PROGRAMMING

A.V. Prutzkow

Ryazan State Radio Engineering University named after V.F. Utkin, Russian Federation, Ryazan, http://prutzkow.com

Abstract. Combinatorics is widely used in computer science and in programming in particular. Combination operations, such as their enumeration, generation, and clustering are performed when programming certain tasks, testing programs, and analyzing algorithms. We collect the most effective ways of performing operations with combinations for further use for scientific and educational purposes.

Keywords: combinatorics, combinations, subset, combination enumeration, generating all subsets.

Введение

Комбинаторика широко используется в информатике [1-2]. При программировании некоторых задач (например, криптографии), при тестировании программ, анализе временно́й сложности алгоритмов возникает необходимость выполнять различные операции с сочетаниями. «Сочетанием из n элементов по m называется любое подмножество из m элементов, которые принадлежат множеству, состоящему из n элементов» [3].

С сочетаниями выполняются следующие операции: нумерация, генерация, а также кластеризация. Каждая из этих операций имеет несколько способов выполнения.

Цель работы – выявить наиболее эффективные способы выполнения операций с сочетаниями в программировании. Критерием выбора способа выполнения являются сложность программной реализации и трудоемкость алгоритма выполнения операции на основе экспертной оценки.

Число сочетаний

Число сочетаний m элементов из n элементов определяется по формуле:

$$C_n^m = \frac{n!}{m!(n-m)!}.$$

В зарубежных публикациях и их переводах на русский язык (например, [4, 5]) сочетания называются -элементными подмножествами. Число сочетаний m по n обозначается как $\binom{n}{k}$, где $k = m$.

Нумерация сочетаний

Каждому сочетанию можно поставить в взаимно-однозначное соответствие номер. Способ нумерации сочетаний [6] включает следующие формулы.

Введем следующие обозначения:

$S = (s_1 s_2 \dots s_n)$ – бинарное представление сочетания, где разряд $s_i = 1$, если в сочетание входит -й элемент, и $s_i = 0$ в противоположном случае, $i = 1,2, \dots, n$.

$x_1 x_2 \dots x_m$ – номера разрядов (слева направо), в которых находится $1,2, \dots, m$-я единица, при этом $x_0 = 0$.

Формула вычисления номера k сочетания в виде $S = (s_1 s_2 \dots s_n)$ имеет вид:

$$k = 1 + \sum_{i=1}^{m} \sum_{j=x_{i-1}+1}^{x_i - 1} C_{n-j}^{m-i}.$$

Получение представления $S' = (s'_1 s'_2 \dots s'_n)$ по номеру k состоит в вычислении значений разрядов по следующим формулам:

$$s'_{i+1} = \begin{cases} 0, \text{если } a_{i+1} > 0, \\ 1, \text{если } a_{i+1} \le 0, \end{cases} \qquad i = 1,2, \dots, n-1,$$

где

$$a_{i+1} = d_i - C_{n-i-1}^{m-l_i-1};$$

$$d_i = \begin{cases} a_i, \text{если } a_i > 0, \\ d_{i-1}, \text{если } a_i \le 0; \end{cases} \qquad l_i = \begin{cases} l_{i-1}, \text{если } a_i > 0, \\ l_{i-1} + 1, \text{если } a_i \le 0; \end{cases}$$

$$d_0 = k, \qquad l_0 = 0, \qquad C_i^0 = 1, \qquad C_i^{-1} = 0.$$

Получение всех сочетаний m по n

Другой операцией с сочетаниями является получение (генерация) их полного списка. Существует несколько алгоритмов получения всех сочетаний m по n (например, в [4, 5, 7]), в том числе и рекурсивных.

Наименьшую сложность программной реализации и трудоемкость алгоритма выполнения операции на основе экспертной оценки имеют приведенные ниже алгоритмы.

Следующая программа реализует алгоритм получения всех сочетаний m по n в лексикографическом порядке, адаптированный из [8]:

```
1  int a[] = new int[m];
2  for (int i = 1; i <= m; i++) {
3        a[i - 1] = i;
4  }
```

```
int p;
if (m == n) {
	p = 1;
} else {
	p = m;
}
while (p > 0) {
	System.out.println(Arrays.toString(a));
	if (a[m - 1] == n) {
		p = p - 1;
	} else {
		p = m;
	}
	if (p > 0) {
		for (int i = m; i >= p; i--) {
			a[i - 1] = a[p - 1] + i - p + 1;
		}
	}
}
```

При разработке итерационных алгоритмов преследуется цель не только сокращения числа итераций, но и сокращения трудоемкости каждой итерации. В работе [9] предлагается один из алгоритмов получения всех сочетаний m по n, имеющий следующие особенности:

– использует бинарное представление сочетаний;

– на каждой итерации меняется только два разряда;

– в теле цикла отсутствуют условия.

Порядок получения сочетаний этим алгоритмом не является лексикографическим.

Адаптированный алгоритм реализуется следующей программой:

```
int[] a = new int[n];
for (int i = 0; i < n; i++) {
	if (i < m) {
		a[i] = 1;
	} else {
		a[i] = 0;
	}
}
int x = m;
int y = m;
System.out.println(Arrays.toString(a));
while (x < n) {
	a[x - 1] = 0;
	a[y - 1] = 1;
	a[0] = a[x];
	a[x] = 1;
	x = x + 1 - (x - 1) * a[1] * (1 - a[0]);
	y = a[0] * y + 1;
	System.out.println(Arrays.toString(a));
}
```

Приведенные выше тексты программ адаптированы под использующиеся в статье обозначения и язык программирования Java. Например, скорректированы индексы элементов массива, так как в языке Java индексы начинаются с 0.

Кластеризация сочетаний

Исходными данными для поиска ассоциативных правил являются транзакции. Транзакции представляют собой сочетания элементов некоторого множества. Выявить закономерности (например, найти похожие по требованиям вакансии [10]) или, наоборот, найти аномалии в данных (например, противоправные действия) помогает кластеризация сочетаний. Алгоритм CLOPE состоит в разбиении сочетаний с максимизацией функции стоимости и имеет преимущества над аналогичными алгоритмами [11].

Заключение

В статье приведены способы выполнения операций: нумерации, получения всех сочетаний m по n и кластеризации, имеющие наименьшую трудоемкость и сложность программной реализации на основе экспертной оценки.

Существуют библиотеки программ, выполняющие эти операции. Однако способ выполнения в библиотеках может быть скрыт. В случае необходимости открытости можно воспользоваться способами приведенными в статье.

Выявленные способы выполнения операций с сочетаниями будут использованы в дальнейших научных исследованиях алгоритмов поиска ассоциативных правил [12], в учебно-методических публикациях и на занятиях по программированию.

Библиографический список

1. Gill Williamson, S. Combinatorics in Computer Science. Computer Science Press, 1985.

2. Lovász, L., Shmoys, D., Tardos, É. Combinatorics in Computer Science, Chapter 40. In R. Graham, M. Grötschel, and L. Lovász (eds.) Handbook of Combinatorics, Elsevier Science B.V., 1995, pp. 2003–2038.

3. Калинина В. Н., Панкин В. Ф. Математическая статистика. – М.: Высш. шк., 1998. – 336 с.

4. Nijenhuis, A., Wilf, H. Combinatorial Algorithms for Computers and Calculators, 2nd ed. Academic Press, 1978.

5. Рейнгольд Э., Нивергельт Ю., Део Н. Комбинаторные алгоритмы. Теория и практика: пер. с англ. – М.: Мир, 1980. – 476 с.

6. Мудров В. И. Алгоритм нумерации сочетаний // Ж. вычисл. матем. и матем. физ. – 1965. – Т. 5. – № 4. – С. 776–778.

7. Кнут Д. Искусство программирования, том 4, А. Комбинаторные алгоритмы, Ч. 1: пер. с англ. – М.: ООО «И.Д. Вильямс», 2013. – 960 с.

8. Липский В. Комбинаторика для программистов: пер. с польск. – М.: Мир, 1988. – С. 40.

9. Ruskey, F., Williams, A. The Coolest Way to Generate Combinations. In Discrete Mathematics, 2009, 309, pp. 5305-5320. DOI: 10.1016/j.disc.2007.11.048.

10. Калевко В. В., Лагерев Д. Г., Подвесовский А. Г. Управление образовательной программой вузов в контексте подготовки конкурентоспособных разработчиков программного обеспечения // Современные информационные технологии и ИТ-образование. – 2018. – Т. 14. – № 4. – С. 803–814. – DOI: 10.25559/SITITO.14.201804.

11. Yang, Y., Guan, H., You, J. CLOPE: A Fast and Effective Clustering Algorithm for Transactional Data. In Proc. of the 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD'02), ACM, New York, NY, USA, 2002, pp. 682–687. DOI: 10.1145/775047.775149.

12. Prutzkow, A. Algorithms and Data Structures for Association Rule Mining and its Complexity Analysis. In the European Proceedings of Social & Behavioural Sciences (EPSBS 2018), 2018, pp. 558–568. DOI: 10.15405/epsbs.2018.11.02.62.

УДК 004.67; ГРНТИ 83.77

МЕТОДЫ И АЛГОРИТМЫ ФОРМИРОВАНИЯ АНСАМБЛЕЙ

О.А. Кельцына
Рязанский государственный радиотехнический университет,
Российская Федерация, Рязань, olga.keltsyna@yandex.ru

Аннотация. В статье рассматриваются идеи одних из самых совсеменных и эффективных методов и алгоритмов формирования ансамблей.
Ключевые слова: интеллектуальный анализ данных, ансамбли алгоритмов, формирование ансамблей, бэггинг, бустинг.

METHODS AND ALGOrITHMS FOr THE FOrMATION OF ENSEMBLES

O.A. Keltsyna
Ryazan State Radio Engineering University,
Russia, Ryazan, olga.keltsyna@yandex.ru

Annotation. The article discusses the ideas of one of the most complete and effective methods and algorithms for the formation of ensembles.
Keywords: data mining, ensembles of algorithms, the formation of ensembles, bagging, boosting.

Для достижения наилучшего результата при решении задач интеллектуального анализа данных часто применяется несколько моделей, в таком случает интересует результат работы не каждой отдельной модели, а результат, который дает совокупность моделей, которая называется ансамблям моделей [1].

В последнее десятилетие ансамбли моделей стали областью активных исследований в машинном обучении, что привело к разработке большого числа разнообразных методов формирования ансамблей.

Сначала при формировании ансамбля необходимо выбрать базовую модель. Ансамбль в целом может рассматриваться как составная модель, состоящая из нескольких отдельных моделей:

1. Ансамбль состоит из базовых моделей одного типа, например только из нейронных сетей и т. д.;

2. Ансамбль состоит из моделей различного типа (деревьев решений, нейронных сетей, регрессионных моделей и т. д.).

Следующий шаг при построении ансамбля моделей – решение, как использовать обучающее множество при формировании ансамля?

Возможно использование следующих подходов:

1. Перевыборка - из обучающего множества извлекается несколько подвыборок, каждая из которых используется для обучения одной из моделей ансамбля. Если ансамбль строится на основе различных типов моделей, то для каждого типа будет свой алгоритм обучения.

2. Использование одного обучающего множества для обучения всех моделей в ансабле.

Затем необходимо решить, как будут комбинироваться рзультаты всех моделей? Обычно используются следующие способы комбинирования:

1. Голосование – выбирается то решение, которое было выдано большинством моделей ансамбля.

2. Взвешенное голосование - в ансамбле одни модели могут работать лучше, а другие хуже, соответственно, к результатам одних моделей доверия больше, а к результатам других меньше. При взвешенном голосовании для моделей ансамбля учитивается уровень доверия (их голос умножается на коэффициент доверия), чтобы учесть уровень достоверности результатов.

3. Усреднение (взвешенное или невзвешенное) - выход всего ансамбля может определяться как простое среднее значение выходов всех моделей [1].

К настоящему времени разработано множество различных методов и алгоритмов формирования ансамблей, среди них наибольшее распространение получили такие методы, как бэггинг, бустинг.

Бэггинг формирует набор моделей, которые комбинируются путем голосования или усреднения. В основе работы бэггинга лежит технология, которая называется «Возмущение и комбинирование».

Модели, которые в процессе обучения изменяют свое состояние в соответствии с обучающим множеством, такие как деревья решений и нейронные сети, неустойчивы: даже замена или удаление одного примера могут привести к существенным изменениям в их состоянии. Иными словами, внося незначительные изменения в обучающих данных, мы всегда будем получать новую модель.

Неустойчивость моделей используется для создания ансамблей моделей с помощью технологии «возмущения и комбинирования». Под возмущением понимается внесение каких-либо изменений (в основном случайного характера), в обучающие данные и построение нескольких моделей на измененных данных с последующим комбинированием результатов [2].

Идея бэггинга заключается в следующем. Сначала на основе исходного множества данных путем случайного отбора формируется несколько выборок. Они содержат такое же количество примеров, что и исходное множество, но набор примеров в этих выборках будет различным (ввиду случайного отбора): одни примеры могут быть отобраны по нескольку раз, а другие – ни разу. Далее на основе каждой выборки строится модель и результаты всех моделей сводятся к одному путем голосования или усреднения результатов. Ожидается, что полученный результат будет наиболее точным, чем результат каждой отдельной модели, построенной на основе исходного набора данных. Обобщенная схема процедуры бэггинга показана на рисунке 1.

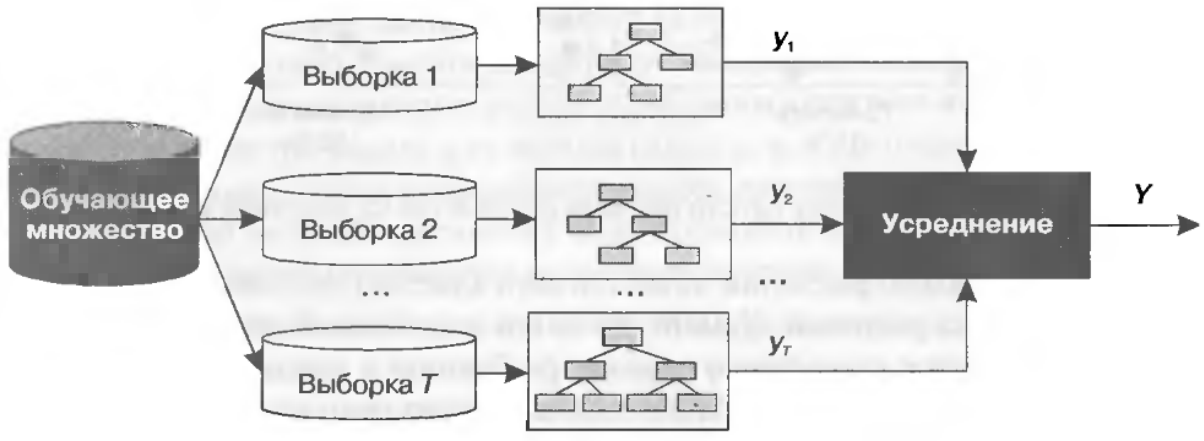


Рис. 1. Схема процедуры бэггинга

Таким образом, бэггинг включает следующие шаги:

1. Из обучающего множества извлекается заданное количество выборок одинакового размера.

2. На основе каждой выборки строится модель.

3. Вычисляется общий результат путем голосования или усреднения всех результатов.

Точность предсказания комбинированных моделей, построенных с помощью бэггинга, оказывается значительно выше, чем точность отдельных моделей.

По сравнению с бэггингом бустинг – более сложный алгоритм, но наиболее эффективный. Как и бэггинг, бустинг использует неустойчивость алгоритмов обучения и создает ансамбль на основе одного исходного множества. Но если в бэггинге модели строятся независимо друг от друга, одновременно, то в бустинге каждая новая модель строится на основе результатов ранее построенных моделей (то есть модели создаются поочередно).

Бустинг создает новые модели таким образом, чтобы они дополняли ранее построенные, выполняли ту работу, которую другие модели сделать не смогли на предыдущих шагах. Так же в отличие от бэггинга в бустинге всем построенным моделям в зависимости от их точности присваиваются веса.

Вместо извлечения выборок из исходного множества данных бустинг в качестве возмущающего фактора применяет взвешивание примеров. Вес каждого примера устанавливается в соответствии с его влиянием на обучение классификатора. На каждой итерации вектор весов подстраивается таким образом, чтобы отражать эффективность данного классификатора. В результате вес неправильно классифицированных примеров увеличивается. И тоговый классификатор также агрегирует обученные классификаторы путем голосования, но теперь голос классификатора является функцией его точности. [1]

Процедура бустинга включает следующие шаги:

1. Для всех примеров исходного множества данных изначально устанавливаются одинаковые начальные веса ω_0;

2. На основе набора примеров строится модель C^i, вычисляется и запоминается выходная ошибка данной модели ε^i;

3. Рассчитывается коррекция весов примеров обучающего множества, веса корректируются согласно формуле:

$$\omega^{i+1} = \omega^i \cdot \frac{\varepsilon^i}{(1-\varepsilon^i)}$$

4, Если ошибка $\varepsilon^i = 0$ или $\varepsilon^i \geq 0{,}5$, то классификатор C^i удаляется и процедура бустинга останавливается;

5. В противном случае осуществляется переход на шаг 2 и начинается следующая итерация.

Таким образом, на каждой итерации происходит настройка параметров (весов примеров). Так наибольший вес будет у примера, который был неправильно распознан наибольшее число раз.

В построении первой модели учавствуют все примеры, в построении второй только те примеры, которые неправильно распознанала первая модель и т. д. Т.е. модели дополняют друг друга и работают с теми примерами, которые предудыщая модель не распознала.

Таким образом, были рассмотрены два современных и эффективных алгоритма формирования ансамбля. Каждый из них обладает рядом преимуществ и предоставляет весьма точные результаты. Однако в виду своей сложности и, соответственно, временных затрат на вычисления эти алгоритмы не всегда могут быть применимы при решении тех или иных задач интелектуального анализа данных.

Библиографический список

1. Паклин Н.Б., Орешков В.И. Бизнес аналитика: от данных к знаниям: Учебное пособие. 2-е изд., испр. - СПб.: Питер, 2013. - 704 с.: ил.
2. Донской В.И. Алгоритмические модели обучения классификации: обоснование, сравнение, выбор. Издательство «ДИАЙПИ», Симферополь, 2014

УДК 004.051; ГРНТИ 50.41.25

ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ АВТОМАТИЗАЦИИ ИССЛЕДОВАНИЯ АЛГОРИТМОВ ОБРАБОТКИ СТРОК

М.А. Лащилин
Рязанский государственный радиотехнический университет,
Российская Федерация, Рязань, mlashchilin@yandex.ru

Аннотация. В работе производится обзор алгоритмов обработки строк и проблемы, связанные с их существующими реализациями. Приводятся классификация алгоритмов, обзор системы их реализации, а также расчет метрик эффективности реализаций алгоритмов.
Ключевые слова: алгоритмы обработки строк, алгоритмы поиска подстроки, выравнивание последовательностей.

INFORMATION SYSTEM FOR AUTOMATION OF RESEARCH OF STRING PROCESSING ALGORITHMS

M.A. Laschilin
Ryazan State Radio Engineering University,
Russia, Ryazan, mlashchilin@yandex.ru

The summary. The paper discusses an overview of string processing algorithms and the problems associated with their existing implementation. Also paper involves the classification of string processing algorithms, the system for their implementation overview and the calculation of performance metrics for the implementation of this algorithms.
Keywords: string processing algorithms, string-search algorithms, sequences alignment.

На данный момент известно большое число алгоритмов обработки строк. Существует их классификация, представленная в книге [1], согласно которой алгоритмы можно разбить на следующие группы: алгоритмы поиска подстроки, алгоритмы приближенного поиска строк, алгоритмы выравнивания строк, алгоритмы синтаксического разбора и т.д. Наиболее часто используемыми являются алгоритмы поиска подстроки. Большая часть языков программирования содержит операции, реализующие поиск строки в тексте. Однако часто это реализация только одного из алгоритмов поиска подстроки. Другие же алгоритмы используются в довольно специфических задачах и, как правило, не входят в стандартный набор операций языка, а их конкретные реализации имеются только в сторонних библиотеках. К тому же не всегда можно получить доступ к исходному коду реализации того или иного алгоритма. Это в свою очередь не позволяет оценить практическое применение алгоритмов.

Следовательно, возникает проблема, заключающая в том, что с теоретической точки зрения правильность алгоритмов доказана, а также оценена их вычислительная сложность, но данные показатели хороши для сравнения алгоритмов между собой и порой не отражают реальной производительности алгоритмов на том или ином языке программирования.

Цель работы заключается в том, чтобы программно реализовать следующие алгоритмы обработки строк:

- алгоритмы поиска подстроки в тексте;
- алгоритмы приближенного поиска строк;
- алгоритмы выравнивания последовательностей;

и затем выполнить их на различных входных данных с целью измерения параметров их работы и на основе выполненных измерений произвести расчет метрик, которые позволят оценить производительность алгоритмов в рамках языка, на котором они реализованы.

Для решения обозначенной проблемы предполагается разработать программное обеспечение на языке программирования Java в соответствии с требованиями, представленными в учебнике [2], которое позволяет выполнять алгоритмы обработки строк (алгоритмы поиска подстроки, алгоритмы приближенного поиска строк, алгоритмы выравнивания строк) при различных входных данных и в процессе выполнения замерять характеристики, которые затем будут применяться при расчете метрик производительности алгоритмов. Выбор Java в качестве языка программирования обусловлен его широким использованием в промышленной разработке.

Предполагается, что разрабатываемое программное обеспечение должно содержать следующие основные модули:

- модуль, который отвечает за исполнение алгоритмов;
- модуль сбора параметров в процессе выполнения алгоритма;
- модуль записи полученных результатов;
- модуль отображения полученных результатов.

Рассмотрим каждый из них. Так, самым важным является модуль, который отвечает непосредственно за исполнение алгоритмов. Именно в этот модуль должны входить непосредственно реализации алгоритмов в виде методов классов, а также классы, которые отвечают за запуск алгоритмов и передачу на их вход требуемых данных.

Модуль сбора параметров в процессе выполнения алгоритма должен фиксировать параметры работы исполняемого в текущий момент алгоритма, которые затем будут использоваться для расчета метрик производительности. Требуется фиксировать следующие параметры:

- время выполнения алгоритма на строках различной длины;
- количество памяти, используемое алгоритмом на строках различной длины;
- время фазы подготовки алгоритма (для алгоритмов, которые предусматривают данную фазу);
- число сравнений, выполняемых алгоритмом (для алгоритмов поиска подстроки);
- средняя величина сдвига в алгоритмах поиска подстроки.

Модуль записи полученных результатов отвечает за запись данных, полученных модулем сбора параметров, в базу данных. Соответственно база данных должна иметь соответствующую структуру, которая бы позволяла хранить для каждого исполнения алгоритма параметры его работы, а также позволяла бы для каждого алгоритма хранить средние значения параметров.

Модуль отображения полученных результатов отвечает за визуализацию данных о работе алгоритмов в наглядном для человека виде. Основными из таких наглядных форм представления данных для человека являются таблицы и графики. Как уже было сказано, это в свою очередь способствует наглядности полученных результатов, а также более простому выявлению зависимостей в данных.

Рассмотрим более подробно структуру базы данных. База данных должна включать в себя следующие таблицы: «Алгоритмы», «Параметры каждого запуска алгоритма», «Метрики».

Таблица «Алгоритмы» содержит всю необходимую информацию о каждом реализованном в системе алгоритме. Данная таблица должна предусматривать возможность хранения названия алгоритма, его краткое описание. Кроме того каждый алгоритм должен иметь

собственный числовой идентификатор, необходимый для построения связей с другими таблицами.

Таблица «Параметры каждого запуска алгоритма» требуется для хранения значений параметров работы алгоритма во время каждого исполнения алгоритма. После каждого исполнения алгоритма в данную таблицу должны заноситься время запуска алгоритма, а также зафиксированные в процессе выполнения параметры.

Таблица «Метрики» необходима для хранения значений метрик эффективности алгоритмов, которые рассчитываются на основе параметров исполнения алгоритмов. Предполагается, что запись в таблицу производится при помощи функциональности расчета метрик после того, как будет выполнено достаточное число исполнений алгоритма, для которого производится расчет метрики.

Перейдем к рассмотрению метрик эффективности работы алгоритмов. Предполагается использовать следующие:

– среднее время выполнения алгоритма на строках одной и той же длины (включая фазу подготовки для алгоритмов, которые её предусматривают):

–

$$t_{полн} = \frac{1}{N}\sum_{i=1}^{N}(t_{под_i} + t_{испол_i}), \quad (1)$$

где $t_{под_i}$ - время выполнения фазы подготовки алгоритма на строках одной и той же длины,

$t_{испол_i}$ - время выполнения основной фазы алгоритма на строках одной и той же длины,

N - число исполнений алгоритмов;

– среднее количество оперативной памяти, используемой в процессе выполнения алгоритмом:

–

$$I_{ср} = \frac{1}{N}\sum_{i=1}^{N} I_i, \quad (2)$$

где I_i - количество оперативной памяти, используемой алгоритмом в процессе выполнения на строках одной и той же длины,

N - число исполнений алгоритмов;

– среднее число сравнений, выполняемых алгоритмом:

–

$$k_{ср} = \frac{1}{N}\sum_{i=1}^{N} k_i, \quad (3)$$

где k_i - число сравнений, произведенных алгоритмом на строках одной и той же длины,

N - число исполнений алгоритмов;

– средняя величина сдвига для N исполнений алгоритма:

–

$$\Delta_{ср} = \frac{1}{N}\sum_{i=1}^{N} \Delta_i, \quad (4)$$

где Δ_i - средняя величина сдвига во время одного исполнения алгоритма на строках одной и той же длины,

N - число исполнений алгоритмов;

– среднее время выполнения алгоритма при различных отношениях длины искомой строки к длине текста (данная метрика вычисляется при значениях отношения длины искомой строки к длине текста, равных $0,01$, $0,10$, $0,15$, $0,20$, $0,30$):

–

$$t_{ср} = \frac{1}{N}\sum_{i=1}^{N} t_i , \quad (5)$$

где t_i - время алгоритма на строках одной и той же длины,

N - число исполнений алгоритмов.

Таким образом, предполагается разработать программную систему, которая позволяет на основе исполнения алгоритмов обработки строк получить параметры их работы. Это в свою очередь позволит оценить реальную эффективность алгоритмов, вычислив на основе параметров рассмотренные метрики, а также выявить возможности для их улучшения с учетом особенностей языка программирования Java. Для выяснения факта улучшения эффективности алгоритмов предполагается производить изменения в их реализациях и затем производить повторный расчет метрик с целью сравнения с ранее полученными результатами.

В дальнейшем предполагается использовать результаты данного исследования для учебного пособия, в котором рассматривались бы данные алгоритмы. Это в свою очередь будет способствовать лучшему пониманию принципов работы строковых алгоритмов студентами.

Библиографический список

1. Crochemore, M., Hancart C., Lecroq, T.. Algorithms on Strings. Cambridge University Press, 2007.

2. Пруцков А. В. Программирование на языке Java. Введение в курс с примерами и практическими заданиями: учебник. — М.: КУРС, 2018. — 208 с.

УДК 378.02:37.016; ГРНТИ 14.35.07

АВТОМАТИЗАЦИЯ ПРОВЕРКИ КАЧЕСТВА ЛАБОРАТОРНЫХ РАБОТ ПО ПРОГРАММИРОВАНИЮ С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТОВ НЕПРЕРЫВНЫХ ИНТЕГРАЦИЙ

И.С. Федюкин

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, me@ifedyukin.ru.

Аннотация. В работе рассматриваются особенности автоматизации процесса проверки качества и соответствия критериям корректности выполнения лабораторных работ по программированию на языке JavaScript с использованием инструментов непрерывных интеграций.

Ключевые слова: непрерывные интеграции, анализ качества кода, автоматизация проверки практических работ, автоматизация проверки лабораторных работ.

AUTOMATION OF QUALITY CONTROL OF PRACTICE PROGRAMMING WORKS USING CONTINUOUS INTEGRATION TOOLS

I.S. Fedyukin
Ryazan State Radio Engineering University named after V.F. Utkin,
Russia Federation, Ryazan, me@ifedyukin.ru.

The summary. The paper discusses the features of automating the process of quality control and compliance with the criteria of correctness of JavaScript programming practice works using continuous integration tools.
Keywords: continuous integrations, code quality analysis, practical works testing automation.

Непрерывная интеграция (CI, англ. Continuous Integration) — практика разработки программного обеспечения, которая заключается в постоянном слиянии рабочих копий в общую основную ветвь разработки (до нескольких раз в день) и выполнении частых автоматизированных сборок проекта для скорейшего выявления потенциальных дефектов и решения интеграционных проблем [1]. Эта методология широко используется в профессиональной разработке программного обеспечения, однако, в процессе обучения программированию преподавателям требуется также выявлять потенциальные дефекты как можно раньше и быстрее. Автоматизация этого процесса сильно ускорит проверку и прием лабораторных работ, а также уменьшит необходимость коммуникаций с каждым студентом для объяснения одинаковых ошибок.

Основы подхода

В основе рассматриваемого подхода лежит подход «Test Driven Development» (TDD), для обучения новым технологиям в области программирования и закрепления изученного материала она хорошо подходит [2]. Часто преподаватели изначально используют этот подход, но без каких-либо дополнительных средств и интеграций.

Изначально преподаватель составляет задание для лабораторной работы и реализует набор тестов для проверки корректности выполнения задания. Ввиду большого количества средств проверки качества кода, в процессе тестирования разработанная студентом программа может быть проверена почти полностью или совсем полностью, причем стоит учесть, что для проверки могут быть использованы не только тесты, но и дополнительные инструменты, измеряющие память, потребляемую программой, время работы или эмулирующие проблемы с сетью или другим внешним модулем.

Таким образом только использование подхода TDD автоматизирует проверку корректности выполнения задания, однако, на начальном этапе требует дополнительных временных вложений от преподавателя, а также накладывает дополнительные ограничения на структуру разрабатываемой программы, в частности, прямо в задании должны быть четко указаны названия классов, в которых должен быть реализован требуемый функционал, а также должны быть четко описаны интерфейсы функций, доступных как внешний API разрабатываемой программы. В случае, если эти условия не будут выполнены, тесты будут давать информация о некорректной работе программы даже, если она на самом деле работает корректно. В качестве инструмента для запуска тестов и сбора статистики по ним может быть использован Jest – инструмент от компании Facebook для запуска тестов.

Помимо проверки корректности работы программы преподаватель проверяет корректность структурного оформления исходного кода, а также правильность его стилистического оформления. Этот аспект также может быть почти полностью автоматизирован с использованием таких инструментов, как статические анализы кода и линтеры: SonarQube и ESlint.

Изначально преподаватель должен задать конфигурации для этих инструментов. Для ESlint конфигурация представляет собой файл в формате JSON, в котором описывается на-

бор необходимых правил и их серьезность. Для SonarQube конфигурация выполняется либо с использованием конфигурационного файла со специфичным форматом, либо через отдельную панель администратора.

С помощью этих инструментов могут быть выявлены ошибки в форматировании кода, его структурном представлении, ошибки в составлении иерархии классов и некорректном использовании модификаторов доступа, некорректные приведения типов и использования ссылочных и простых типов, также во всем коде программы находятся дублирования кода, фрагменты с высокой цикломатической сложностью, потенциальные неверные выходные условия рекурсии, а также другие частые ошибки, которые могут быть выявлены на этапе статического анализа. Помимо простого выявления ошибок эти средства также позволяют собирать статистику.

Технические особенности

В обучении внутри компаний все чаще используют практику непрерывных интеграций [3] для автоматического запуска проверок, рассмотренных ранее. В основе непрерывных интеграций лежат два модуля: система хранения исходного кода и сервер непрерывных интеграций. Общий алгоритм работы непрерывных интеграций изображен на рисунке 1.

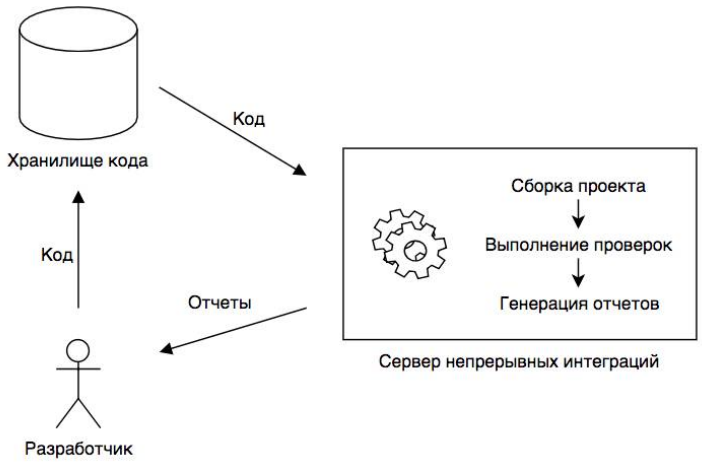


Рис. 1. Общая схема работы механизма непрерывных интеграций

В качестве хранилища кода может быть использован Gitlab – этот сервис предоставляет бесплатную возможность использования приватных репозиториев, а также имеет встроенный сервис непрерывных интеграций – Gitlab CI.

Изначально преподаватель создает набор git-репозиториев, каждый репозиторий соответствует одному варианту заданий, каждая ветка соответствует одной лабораторной работе по теме.

В репозиториях содержатся шаблон для выполнения лабораторной работы, конфигурационные файлы для ранее рассмотренных инструментов (SonarQube, ESlint, Jest), набор тестов для проверки корректности выполнения задания, непосредственно текст задания и методические указания в формате Markdown, а также описание пайплайна непрерывных интеграций.

Таким образом в одном репозитории содержится все необходимое для изучения темы, выполнения лабораторной работы и ее автоматической проверки. Студент создает форк репозитория и таким образом получает свою приватную копию репозитория и может приступать непосредственно к выполнению лабораторной работы без необходимости дополнительных настроек или коммуникаций с преподавателем.

После выполнения лабораторной работы и отправки изменений в репозиторий Gitlab автоматически запустит процесс проверки лабораторной работы, общая схема которого представлена на рисунке 2.

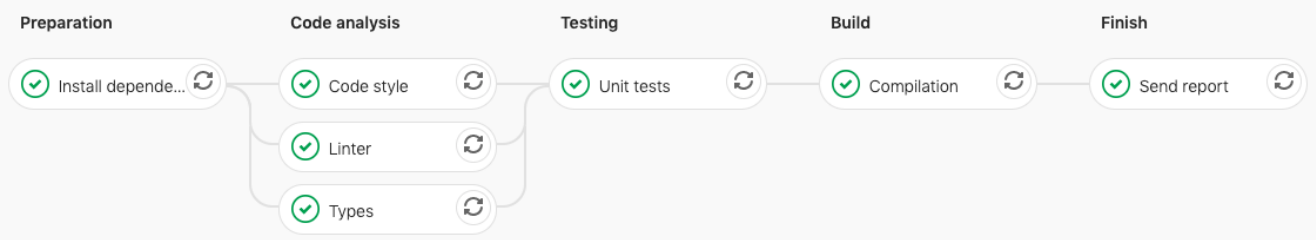


Рис. 2. Схема пайплайна непрерывных интеграций

Сначала Gitlab CI разворачивает окружение для проекта и устанавливает необходимые зависимости, к ним относятся библиотеки и инструменты для запуска тестов и линтеров, после чего выполняется статический анализ кода: параллельно запускается ESlint, SonarQube и проверка типов с использованием Flow-анализатора. Далее следуют запуск тестов и сборка проектов. Каждый из шагов в качестве результата предоставляет некоторый отчет о выполнении тех или иных проверках в формате JSON, а также отметку об успешности прохождения проверки.

На последнем шаге выполняется анализ результатов выполнения каждого шага, если какой-то шаг не был пройден успешно, сообщение об этом отправляется студенту-автору изменения, причем к сообщению прикладывается отчет об ошибке, которая возникла для того, чтобы студент имел явное указание на причину возникновения ошибки.

Если все шаги пройдены успешно, то сообщение отправляется преподавателю для того, чтобы он смог приступить к проверке лабораторной работы. Так как репозиторий студента приватный, то вместе с отправкой сообщения выполняется проверка, имеет ли преподаватель доступ к репозиторию студента, если доступа нет, то для аккаунта преподавателя предоставляется доступ на чтение содержимого репозитория. Таким образом преподаватель может завершить проверку лабораторной работы и выполнить обзор кода с использованием встроенных инструментов Gitlab, причем студент будет видеть комментарии преподавателя в режиме реального времени.

Таким образом, с помощью средств непрерывных интеграций, методологии TDD и статического анализа исходного кода программного обеспечения процесс проверки лабораторных работ по программированию может быть автоматизирован почти полностью.

Библиографический список

1. Wikipedia, Непрерывная интеграция. – https://ru.wikipedia.org/wiki/Непрерывная_интеграция, дата обращения: 14.02.2020.

2. Proceedings of the 13th annual conference on Innovation and technology in computer science education, Test-driven development in education: experiences with critical viewpoints. – https://dl.acm.org/citation.cfm?id=1597849.1384306, дата обращения: 15.02.2020.

3. Erik Kral, Petr Capek. Towards Using Continuous Integration Tools to Teach Programming Courses: сборник публикаций конференции. – М.: 2015 International Conference on CSCI, 2016 – 1-2 с.

УДК 004.67; ГРНТИ 83.77

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЗАДАЧ КЛАССИФИКАЦИИ ПРОГНОЗИРОВАНИЯ

О.А. Кельцына
Рязанский государственный радиотехнический университет,
Российская Федерация, Рязань, olga.keltsyna@yandex.ru

Аннотация. В статье рассматриваются задачи классификации и прогнозирования, которые имеют ряд сходств и порой их путают между собой. Рассмотрена каждая задача по-отдельности и представлен итоговой сравнительный анализ двух задач.
Ключевые слова: классификация, прогнозирование, интеллектуальный анализ данных, построение модели классификации, модель прогнозирования.

COMPARATIVE ANALYSIS OF PROBLEMS FOR CLASSIFICATION OF FORECASTING

O.A. Keltsyna
Ryazan State Radio Engineering University,
Russia, Ryazan, olga.keltsyna@yandex.ru

Annotation. The article discusses the problems of classification and forecasting, which have a number of similarities and sometimes confuse them. Each task is considered separately and the final comparative analysis of the two tasks is presented.
Keywords: classification, forecasting, data mining, construction of a classification model, forecasting model.

Многие методы интеллектуального анализа данных (Data Mining) используются для решения различных задач классификации и прогнозирования. Это, например, нейронные сети, линейная регрессия, деревья решений (которые иногда называют - деревья прогнозирования и классификации). Задачи прогнозирования и классификации имеют сходства и различия. Рассмотрим их подробнее и приведем сравнительный анализ этих задач.

Задача классификации одна из наиболее частых задач интеллектуального анализа данных, она возникает практически во всех областях человеческой деятельности. Классификация – это отнесение по определенному принципу объектов к различным классам. У каждого объекта есть набор свойств (характеристик), которые отличают его от других объектов и по которым производится отнесение объекта к определенному классу. В задаче классификации все классы известны заранее.

Классификация относится к стратегии обучения с учителем, которое также именуют контролируемым или управляемым обучением. Часто задачу классификации относят к задаче предсказания категориальной зависимой переменной на основе выборки непрерывных или категориальных переменных.

Например, можно предсказать, какой из писем в почтовом ящике является спамом, а какое - нет, какое письмо следует перенести в папку со спамом, а какое оставить в папке со всеми входящими письмами. Приведенный пример задачи относится к задачам бинарной классификации, в которых зависимая переменная может принимать только два значения – да или нет (1 или 0). Так же существует вариант мультиклассовой классификации, когда зависимая переменная может принимать значения из некоторого множества заранее определенных классов. Например, когда необходимо предсказать, какой фирмы захочет купить клиент технику в магазине. В этих случаях рассматривается множество классов для зависимой переменной. Классификация может быть одномерной, когда каждый объект классифицируется по одному признаку, и многомерной, когда объекты классифицируются по двум и более признакам. Рассмотрим задачу классификации на простом примере. Допустим, имеется база данных о клиентах банка с информацией о возрасте и доходе за месяц. Есть два рекламных

предложения: кредит на большую сумму и кредит на небольшую сумму. Соответственно, определены два класса клиентов: класс 1 и класс 2 [1].

Для наглядности представим базу данных банка из примера в двухмерном измерении (доход и возраст), в виде множества объектов, принадлежащих классу 1 (кружки с надписью «1») и классу 2 (кружки с надписью «2»). На рисунке 1 приведены объекты из двух классов.

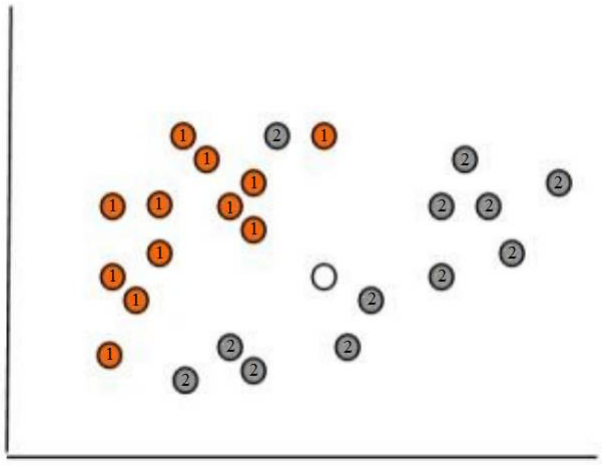


Рис. 1. Множество объектов классов

Задача классификации заключается в том, чтобы определить, к какому классу (1 или 2) принадлежит новый клиент (на рисунке 1 – белый кружок) и какой из двух видов предложений ему стоит отсылать.

Таким образом, определение класса, к которому относится клиент, является решением задачи классификации, а прогнозирование дохода, который принесет этот клиент банку в будущем году, будет решением задачи прогнозирования. Теперь рассмотрим подробнее задачу прогнозирования.

Задачи прогнозирования решаются в самых разнообразных областях человеческой деятельности, науке, экономике, производстве и множество других сфер. Под прогнозированием понимают научное исследование, которое направлено на то, чтобы выявить перспективы развития явления или процесса.

Задача прогнозирования является одной из самых сложных задач интеллектуального анализа данных, она требует тщательного исследования исходного набора данных, а также методов, подходящих для их анализа.

Целью прогнозирования является предугадывание состояния объекта и будущих событий на основе ретроспективных данных. Под ретроспективными данными объекта понимают его состояния в прошлом и настоящем. Таким образом, как и для задачи классификации, чтобы решить задачу прогнозирования требуется некоторая обучающая выборка данных. При решении задачи прогнозирования устанавливается функциональная зависимость между зависимыми и независимыми переменными.

Главная функция прогнозирования состоит в проведении анализа различных экономических и социальных тенденций, а также в предсказании новых экономических явлений и выявлении возможных экономических проблем [2].

Прогнозирование позволяется уменьшить риск принятия необоснованные и неверных решений. Примеры его задач: прогноз финансовых перспектив предприятия, прогноз урожайности определенных агрокультур, прогноз движения денежных средств, прогнозироваание продаж товаров и т.д.

Помимо экономической и финансовой сферы, задачи прогнозирования решаются в самых разнообразных областях: политике, фармакологии, медицине и т.д. В общих чертах решение для решения задачи прогнозирования можно выделить следующие подзадачи:

• выбор модели прогнозирования;

• анализ адекватности и точности построенного прогноза [2].

Таким образом, задача прогнозирования схожа с задачей классификации. Многие методы интеллектуального анализа данных используются как для решения задач классификации, так и для задач прогнозирования (например, деревья решений, нейронные сети, метод опорных векторов, случайный лес и т.д) Основное сходство задачи классификации и прогнозирования заключается в том, что при решении обеих задач происходит двухэтапный процесс построения модели. Модель строится на основе обучающего набора данных. Она используется для предсказания неизвестных значений зависимой переменной. Основным различием задач классификации и прогнозирования является то, что в задаче классификации предсказывается класс зависимой переменной, а в задаче прогнозирования - числовые значения зависимой переменной, пропущенные или неизвестные (которые относятся к будущему).

Библиографический список

1. И.А. Чубукова. Data Mining : учебное пособие / - 2-е изд., испр. - Москва : Интернет - Ун-т Информ. Технологий : БИНОМ. Лаб. знаний, 2008. - 382 с. : ил., табл.;

2. Новикова Н.В., Поздеева О.Г. Прогнозирование национальной экономики: Учебно-методическое пособие. Екатеринбург: Изд-во Урал. гос. экон. ун-та, 2007;

УДК 004.421; ГРНТИ 50.41.25

ПОДХОДЫ ПОВЫШЕНИЯ ТОЧНОСТИ И МЕТОДИКИ АНАЛИЗА РЕЗУЛЬТАТОВ САМОСТОЯТЕЛЬНОГО ТЕСТИРОВАНИЯ ЗРЕНИЯ С ПОМОЩЬЮ КОМПЬЮТЕРНОЙ СИСТЕМЫ

А.В. Благодаров, Н.А. Тярт

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, tyartnikita@gmail.com

Аннотация. В работе рассматриваются особенности реализации программного обеспечения для самостоятельного тестирования состояния зрения пользователей без специальной подготовки. Описываются основные подходы для снижения погрешностей измерения. Приводятся методики анализа накапливаемых результатов в краткосрочной и долгосрочной перспективах.

Ключевые слова: проверка зрения, самотестирование, программная система, точность измерений, анализ данных.

APPROACHES TO IMPROVE THE ACCURACY AND METHODS OF ANALYZING THE RESULTS OF SELF VISION TESTING USING A COMPUTER SYSTEM

A.V. Blagodarov, N.A. Tyart

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, tyartnikita@gmail.com

The summary. The paper discusses the features of implementing software for self-testing the state of users vision without special qualification. The main approaches for reducing measurement errors are described. Methods of analyzing accumulated results in the short and long term are presented.

Keywords: vision check, self-testing, software system, measurement accuracy, data analysis.

Концепция

Активное развитие и массовое потребление цифровых технологий широкими слоями пользователей, а в частности – связанное с этими технологиями использование дисплеев с активным излучением света, имеющих зачастую малые размеры, влечет за собой увеличение частоты возникновения проблем со зрением. Из этого следует необходимость повышенного внимания каждого пользователя подобных электронных устройств к состоянию своего зрения.

Согласно различным рекомендациям, для большинства людей в рамках профилактики заболеваний глаз необходимо посещать врача-офтальмолога с периодом в 1-2 года. С учетом вышеописанной проблемы такой частоты проверок явно недостаточно. Однако нельзя просто дать всем рекомендации об увеличении частоты посещения врача, например, в 2 раза. Во-первых, многие и сейчас не следуют подобным рекомендациям, поэтому их изменение не возымеет должного эффекта. Во-вторых, если получится увеличить частоту посещения врачей всеми людьми, то это просто вызовет черезмерное повышение нагрузки на специалистов и большие очереди в медицинских учереждениях.

Выходом из описанной ситуации является нахождение достаточно эффективных способов самопроверки зрения, которые могут быть использованы большинством пользователей при минимальном количестве временных затрат и с приемлемой частотой проведения. При этом стоит понимать, что ни одна методика самостоятельного тестирования не приблизится по точности и комплексности к обследованию у профессионального врача-офтальмолога. Следовательно, самотестирование должно стать лишь вспомогательной мерой, позволяющей увеличить частоту контроля состояния здоровья глаз, а не полноценной заменой профессиональному медицинскому обследованию.

Один из самых простых, но в то же время и самых эффективных способов реализации методик самотестирования – создание программного обеспечения для современных цифровых устройств. Такое средство должно объяснять и организовывать процесс тестирования, а также хранить и анализировать получаемые результаты измерений. Согласно исследованиям [1], с помощью подобного программного обеспечения возможно обеспечить приемлемую точность измерений и получить полезные результаты.

На текущий момент существует множество программных средств для проведения различных видов тестирования состояния зрения [2, 3]. Они реализованы в виде интернет-сервисов, приложений для мобильных устройств и стационарных компьютеров. Но все эти системы работают по принципу единовременных замеров и пытаются сразу после одного тестирования выдать результат в абсолютных величинах (например, подобная система может выдать, что острота вашего зрения равна 0.6, или 60% от нормы). Более того, большинство представленных решений не использует никакие подходы для устранения (очевидных и очень больших) погрешностей измерения, связанных с особенностями реализации и типичной логикой использования подобных систем.

Исходя из рассмотернных выше задач и проблем, далее описыватеся предлагаемое программное обеспечение, находящееся на момент написания статьи в процессе разработки. Основная идея создаваемого решения – организация периодического самостоятельного тестирования пользователями состояния зрения. Это позволит не только повысить точность измерений за счет компенсации шумов и периодических колебаний состояния зрения, но и производить анализ динамики зрения, строить прогнозы и предупреждать пользователей о найденных опасных тенденциях. Разработка ведется для ОС Android и представляет собой типичное пользовательское приложение, включающее в себя несколько видов тестов, главным из которых является тест остроты зрения. Далее рассматриваются особенности реализации в первую очередь именно этого вида теста, включая подходы к анализу собираемых приложением результатов измерения, дающие конкуретное преимущество перед аналогичными решениями.

Особенности процесса тестирования

Говоря про учитывание особенностей формата проведения теста, в первую очередь стоит рассмотреть аспект аппаратной платформы, на которой выполняется тестирование. ОС Android установлена на очень большом количестве разнообразных устройств, у которых отличаются и физические параметры дисплеев, и программные конфигурации системы. Для корректной организации проведения теста остроты зрения, в ходе которого подразумевается вывод на дисплей устройства символов определенного размера и на определенном расстоянии от пользователя, необходимо учитывать и программные, и аппаратные параметры целевого устройства.

ОС Android частично поддерживает автоматическое масштабирование изображение через механизм указания размеров элементов интерфейса относительно параметра DPI (dots per inch, точек на дюйм) [4]. Но данный подход не гарантирует точного соответствия виртуальных и реальных размеров. Более того, зачастую производители устройств сами указывают неверное значение DPI для различных моделей, чтобы изменить масштаб интерфейса во всей системе по своему разумению.

Единственно возможным выходом является ручное указание физических размеров дисплея устройства пользователем. Эти данные можно запросить различными способами, но в результате всё равно пользователю понадобится взять в руки линейку и измерить дисплей либо его часть, после чего ввести результаты измерений в приложение. Только после этого можно будет гарантировать корректное масштабирование изображения относительно дисплея устройства.

Второй проблемой масштабирования является расстояние между дисплеем и глазом пользователя. Стандартный тест проверки остроты зрения проводится с расстояния в 5 метров, но данная величина неприменима для тестирования при помощи дисплея смартфона/планшета. Чаще всего тестирование будет проводится с расстояния вытянутой руки, это максимальное расстояние, которое пользователь приложения сможет обеспечить без дополнительных приготовлений. Следовательно, требуется ввод в приложение параметра длины руки пользователя. Но если пользователь не может в данный момент измерить длину своей руки напрямую, то можно вычислить ее приблизительно как половину роста человека (а свой рост подавляющее большинство людей хорошо знает).

Стандартным набором символов для проведения теста остроты зрения на данный момент является таблица Головина–Сивцева, использующая символы «Ш», «Б», «М», «Н», «К», «Ы» и «И». Но она имеет существенный недостаток, заключающийся в различном размере промежутков между элементами символа, что вводит дополнительные погрешности в результаты тестирования. В сторонних исследованиях [5] предлагается использование альтернативных наборов символов, лишенных этого недостатка.

Известным фактом является то, что острота зрения ухудшается к концу дня из-за усталости глаз. Однако то, насколько силен этот эффект, индивидуально для каждого человека. Выходом в данной ситуации будет нахождение средней величины падения остроты зрения, о чем будет рассказано подробнее в следующем разделе статьи. Для этого необходимо не только засекать точное время снятия результатов по часам устройства, но и в явной форме просить пользователей вводить данные о том, в какой части активного дня (начало, середина или конец) проводилось тестирование. Для большинства пользователей эти варианты будут соответствовать утру, дню и вечеру светового дня, но не стоит забывать о пользователях, живущих по ночному или даже скользящему графику, что серьезно усложнит анализ данных без их ручной разметки.

Последним предлагаемым вариантом добавления данных в систему является ручной ввод результатов тестирования у врача-офтальмолога. Таким результатам должен отдаваться приоритет при определении поправок к полученным результатам самотестирования.

Анализ результатов тестирования

Первый этап анализа накапливаемых в системе результатов тестирования заключается в поиске и компенсации различных шумов и периодических колебаний. Всё это направлено на повышение точности оценки текущей динамики состояния зрения и увеличение правдоподобности экстраполяции результатов для прогнозирования дальнейших изменений здоровья пользователя. Здесь предполагается использование сразу нескольких подходов, учитывающих и моды, и средние арифметические значения показаний за определенные части активного дня / день / неделю / месяц, а также установление порогов для отсеивания шумов (неправдоподобно больших отклонений единичных результатов замеров от соседних значений).

Если график активности пользователя не является плавающим (его начало, середина и конец активного дня всегда начинаются и заканчиваются примерно в одно и то же время), то можно ввести линейно-кусочную функцию поправки для приведения результатов замеров к некоторому усредненному показателю. Эта функция будет постоянно калиброваться системой под конкретного пользователя с добавлением новых результатов, но стоит учесть, что для ее построения желательно не использовать слишком старые данные (старее 3-6 месяцев). В случае с плавающим графиком активности ввод такой функции невозможен, но в качестве альтернативы можно использовать три константные поправки для замеров, сделанных в начале, середине и конце активного дня соответственно. Эти константы можно вычислять по аналогичным алгоритмам. Все эти поправки должны иметь дополнительное смещение в сторону соответствия результатам профессиональной проверки зрения у врача-офтальмолога.

После сбора массива результатов самотестирования за определенный промежуток времени можно проводить экстраполяцию функции динамики состояния зрения на ближайшее будущее с целью выдачи прогнозов пользователям. Линейная экстраполяция подойдет далеко не для всех случаев, поэтому может потребоваться использование более сложного математического аппарата.

Анализ результатов тестирования можно проводить в различных срезах. Оценку динамики состояния имеет смысл рассчитывать как в краткосрочной (1-2 недели), так и в долгосрочной перспективе (1 месяц и более). В случае обнаружения отклонений в худшую сторону необходимо сразу же предупредить об этом пользователя в достаточно явном виде, чтобы он не пропустил случайно сообщение системы, и дать рекомендацию внепланового посещения врача. При получении неправдоподобных результатов измерений, выбивающихся из общего ряда, имеет смысл предложить пользователю сразу же повторить тестирование, по возможности устранив перед этим все факторы, мешающие корректному ходу тестирования.

Выводы

Программная система для самостоятельной проверки зрения, описанная в этой работе, направлена на повышение частоты контроля состояния здоровья глаз простых пользователей. Она дает возможность уменьшить задержки между возникновением различных заболеваний глаз и началом их лечения, что зачастую позволяет уменьшить вред здоровью и избежаать побочных осложнений. Накопление и обработка результатов измерений дают возможность производить анализ динамики здоровья глаз, прогнозировать его дальнейшее изменение и более точно определять текущее состояние, производя калибровку в том числе и под конкретного пользователя.

Библиографический список

1. Кладов, Г.К. Апробация компьютерных тестов для проверки зрения / Г.К. Кладов, А.С. Винокуров, Л.В. Подригало, Т.Ю. Мителева // Вестник НТУ «ХПИ». Сборник научных трудов. Тематический выпуск: Информатика и моделирование. – 2004. – № 34. – С. 73-78.

2. Тесты для проверки зрения – онлайн, бесплатно [Электронный ресурс]. – Режим доступа: https://excimerclinic.ru/eye-tests/, свободный. – Дата доступа: 01.02.2020.

3. Проверка остроты зрения [Электронный ресурс]. – Режим доступа: http://checkeye.ru/ostrota.php, свободный. – Дата доступа: 01.02.2020.

4. Supporting Different Densities | Android Developers [Электронный ресурс]. – Режим доступа: https://developer.android.com/training/multiscreen/screendensities, свободный. – Дата доступа: 05.02.2020.

5. Патраль, А.В. Проверка остроты зрения / А.В. Патраль // Евразийское научное объединение. – 2017. – № 11. – С. 38-41.

УДК 004.8; ГРНТИ 28.23.20

ВИЗУАЛИЗАЦИЯ ДАННЫХ В ЗАДАЧЕ КЛАСТЕРИЗАЦИИ ВРЕМЕННЫХ РЯДОВ

Л.А. Демидова[*,], М.А. Степанов[**]**

[*]Российский технологический университет – МИРЭА, Российская Федерация, Москва;
[**]Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, liliya.demidova@rambler.ru, smastefan@gmail.com

Аннотация. Рассматриваются аспекты визуализации данных в задаче кластеризации временных рядов с применением t-SNE- и UMAP-алгоритмов с целью выявления оптимального числа кластеров, присутствующих в данных, и выбора алгоритма кластеризации.
Ключевые слова: визуализация, временные ряды, кластеризация, t-SNE, UMAP.

DATA VISUALIZATION IN THE PROBLEM OF THE TIME SERIES CLUSTERING

L.A. Demidova[*,], M.A. Stepanov[**]**

[*]Russian Technological University – MIREA, Russia, Moscow;
[**]Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, liliya.demidova@rambler.ru, smastefan@gmail.com

The summary. The aspects of the data visualization in the problem of the time series clustering using the t-SNE- и UMAP-algorithms with the aim of finding of the optimal number of clusters, which are present in the data, and choice of the clustering algorithm.
Keywords: visualization, time series, clustering, t-SNE, UMAP.

Решение задачи кластеризации всегда предполагает выбор алгоритма кластеризации: от выбора алгоритма зависит качество полученного разбиения объектов на кластеры, оцениваемое с применением того или иного показателя качества кластеризации, например, с применением индекса кластерного силуэта, который должен быть максимизирован [1 – 5]. При этом необходимо выбрать не только алгоритм кластеризации, попытавшись угадать структуру как самого набора данных в целом, так и каждого кластера в отдельности, но и предположить, какое число кластеров содержится в анализируемом наборе данных. Каждый алгоритм кластеризации реализует конкретные жесткие математические принципы разбиения объектов на кластеры: даже при работе с одним алгоритмом кластеризации существенную роль могут сыграть используемые в нем метрика расстояния и метод группирования объектов в кластеры. Например, для кластеров гиперэллиптической формы может оказаться целесообразным использование других метрик, чем в случае кластеров гиперсферической формы.

Для получения адекватных результатов кластеризации необходимо исследовать результаты, полученные с применением различных алгоритмов кластеризации при разном числе кластеров с реализацией в этих алгоритмах различных метрик расстояний и других настраиваемых параметров. Такой перебор алгоритмов кластеризации связан с существенными временными затратами. Очевидно, что сокращение перечня исследуемых алгоритмов кластеризации позволило бы сократить временные затраты на получение приемлемого решения, однако ввиду высокой размерности анализируемых наборов данных их визуализация в ис-

ходном метрическом пространстве не представляется возможной. При этом очень остро при решении задачи кластеризации ощущается проблема плохой отделимости кластеров друг от друга: возможно, данные таковы, что попытка их кластеризации вообще не имеет смысла.

Еще более сложной является задача кластеризации группы временных рядов (ВР) [6 – 9], в том числе потому, что приходится работать с данными, имеющими существенно различные диапазоны принадлежности своих значений.

Подход к визуализации

В последние годы для решения задачи визуализации данных большой размерности предлагаются различные алгоритмы, позволяющие представить данные в 2-х или в 3-мерном метрическом пространстве с сохранением пропорциональности расстояний между объектами и структуры кластеров. Варьируя значения параметров таких алгоритмов можно учесть при отображении данных их локальные или глобальные характеристики.

В качестве таких алгоритмов целесообразно назвать, в первую очередь, t-SNE- и UMAP-алгоритмы, программные реализации которых есть в языке Python [10, 11].

t-SNE-алгоритм (t-distributed Stochastic Neighbor Embedding, стохастическое вложение соседей с t-распределением) [10] – алгоритм машинного обучения, который реализует нелинейное снижение размерности и предназначен для визуализации данных высокой размерности посредством вложения их в метрическое пространство низкой размерности (двух- или трехмерное). t-SNE-алгоритм позволяет представить объект высокой размерности двух- или трёхмерной точкой таким образом, что похожие объекты представляются близко расположенными точками, а непохожие объекты представляются точками с большой вероятностью далеко расположенными друг от друга.

t-SNE-алгоритм сначала создаёт распределение вероятностей по парам объектов в метрическом пространстве высокой размерности таким образом, что похожие объекты выбираются с большой вероятностью, в то время как вероятность выбора непохожих объектов невелика. Затем t-SNE-алгоритм находит похожее распределение вероятностей по точкам в метрическом пространстве малой размерности и минимизирует расстояние Кульбака-Лейблера между двумя распределениями с учётом положения объектов. При этом t-SNE-алгоритм решает задачу минимизации с помощью классического градиентного спуска.

Основными параметрами t-SNE-алгоритма являются: перплексия (по умолчанию – 30), параметр раннего преувеличения (по умолчанию – 12), скорость обучения (по умолчанию – 200), число итераций оптимизации (по умолчанию – 1000), максимальное число итераций без прогресса перед прерыванием оптимизации (по умолчанию – 300); пороговое значение нормы градиента (по умолчанию – 1E – 7), метрика (по умолчанию – евклидова); метод вычисления градиента (по умолчанию – 'barnes_hut'), пороговый угол (по умолчанию – 0.5); число параллельных заданий для поиска соседей (по умолчанию – 1) ; размерность конечного метрического пространства (по умолчанию – 2).

Перплексию (perplexity) соотносят с числом ближайших соседей, которое используется в других алгоритмах машинного обучения.

Параметр раннего преувеличения (early exaggeration) управляет тем, насколько плотные кластеры из исходного метрического пространства находятся во встроенном метрическом пространстве и каково будет расстояние между ними. Большие значения этого показателя позволяют установить большее расстояние между кластерами во встроенном метрическом пространстве.

Число итераций определяет максимальное число итераций t-SNE-алгоритма.

Максимальное число итераций без прогресса перед прерыванием оптимизации используемое после 250 начальных итераций с ранним преувеличением.

Если норма градиента ниже порогового значения, оптимизация будет остановлена.

Метрика определяет, каким образом будут вычисляться расстояния в исходном метрическом пространстве.

Метод вычисления градиента по умолчанию выбирается как метод, использующий приближение Барнса-Хата, работающее за время O (n·log(n)). Время работы точного метода вычисления градиента оценивается как $O(n^2)$, где n – число объектов (при этом точный метод не может масштабироваться в случае больших наборов данных (размером около 1 млн.).

Пороговый угол используется только, если метод вычисления градиента – 'barnes_hut'. Он позволяет обеспечить компромисс между скоростью и точностью этого метода и реализует оценку углового размера удаленного узла, измеренного от объекта. Если этот угловой размер меньше порогового угла, он используется как итоговый узел всех объектов, содержащихся в нем. Пороговый угол, меньший чем 0.2, быстро увеличивает время вычислений, а пороговый угол, больший чем 0.8, быстро увеличивает погрешность вычислений.

При увеличении числа параллельных заданий для поиска соседей можно осуществить распарраллеливание t-SNE-алгоритма.

UMAP-алгоритм (Uniform Manifold Approximation and Projection, приближение и проекция равномерного многообразия) [11] – алгоритм машинного обучения, который также реализует нелинейное снижение размерности. Сначала UMAP-алгоритм построение взвешенного графа посредством соединения ребрами только тех объектов, которые являются ближайшими соседями. Множество, составленное из ребер графа, представляет собой нечёткое множество с функцией принадлежности, которая вычисляется как вероятность существования ребра между двумя вершинами. Затем UMAP-алгоритм создает граф в низкоразмерном пространстве и приближает его к исходному посредством минимизации суммы дивергенций Кульбака-Лейблера для каждого ребра из множества ребер. При этом UMAP-алгоритм решает задачу минимизации с помощью стохастического градиентного спуска.

UMAP-алгоритм основан на трех предположениях о данных: данные равномерно распределены на римановом многообразии, представляющем собой вещественное дифференцируемое многообразие, в котором каждое касательное пространство снабжено скалярным произведением (метрическим тензором), меняющимся от точки к точке гладким образом; риманова метрика локально постоянна (или может быть аппроксимирована как таковая); риманово многообразие локально связно.

С учетом этих предположений моделируется многообразие с нечеткой топологической структурой. При этом искомое вложение определяется посредством поиска низкоразмерной проекции данных, которые имеют наиболее близкую эквивалентную нечеткую топологическую структуру.

Основными параметрами UMAP-алгоритма являются: число ближайших соседей (по умолчанию – 15), минимальное расстояние, на котором могут находиться точки в новом метрическом пространстве (по умолчанию – 0.1), метрика (по умолчанию – евклидова); размерность конечного метрического пространства (по умолчанию – 2).

Варьируя число ближайших соседей, можно определить, что важнее сохранить в новом метрическом пространстве: глобальную или локальную структуру данных. Малые значения этого параметра означают, что при оценке исходного метрического пространства UMAP-алгоритм ограничивается малой окрестностью вокруг каждого объекта, пытаясь сохранить локальную структуру данных (может быть, даже в ущерб общей картине). Большие значения *n* этого параметра означают, что при оценке исходного метрического пространства UMAP-алгоритм учитывает точки в большей окрестности, пытаясь сохранить глобальную структуру данных (может быть, упуская детали).

Варьируя величину минимального расстояния, можно определить, в каком виде должны быть представлены данные в новом метрическом пространстве. Малые значения этого параметра позволяют оценить, на какие кластеры разделяются ваши данные. Большие значения этого параметра позволяют увидеть структуру данных, как единое целое.

Метрика определяет, каким образом будут вычисляться расстояния в исходном метрическом пространстве.

При определении параметра, отвечающего за размерность конечного метрического пространства, можно определить размерность уменьшенного пространства, в которое будут встраиваться исходные данные. В отличие от некоторых других алгоритмов визуализации, таких как, например, t-SNE-алгоритм, UMAP-алгоритм, хорошо масштабируется при встраивании, поэтому его можно использовать не только для визуализации в 2-х или 3-х мерном пространствах.

Считается, что UMAP-алгоритм похож на t-SNE, но имеет более сильное математическое обоснование.

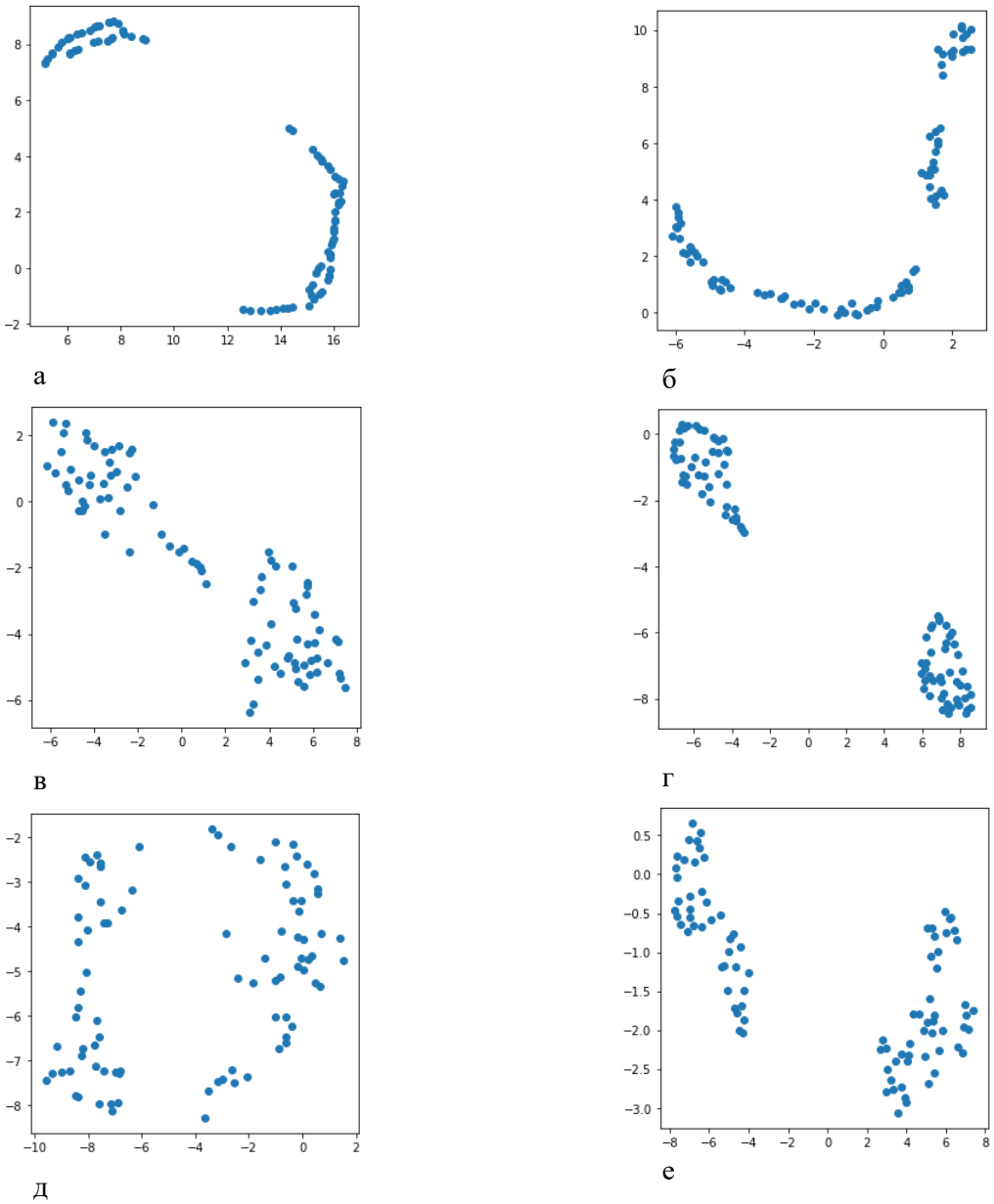


Рис. 1. Визуализация групп ВР:
а – t-SNE-алгоритм при числе отсчетов, равном 3;
б – UMAP алгоритм при числе отсчетов, равном 3;
в – t-SNE-алгоритм при числе отсчетов, равном 4;
г – UMAP-алгоритм при числе отсчетов, равном 4;
д – t-SNE-алгоритм при числе отсчетов, равном 5;
е – UMAP-алгоритм при числе отсчетов, равном 5

При использовании t-SNE- и UMAP-алгоритмов можно попытаться выполнить визуализацию данных (например, временных рядов), и выдвинуть предположение о целесообразности работы с уже ограниченным списком алгоритмов кластеризации. Кроме того, можно выдвинуть предположение и об оптимальном числе кластеров, присутствующих в анализируемом наборе данных. При этом, варьируя значения параметров t-SNE- и UMAP-алгоритмов, можно улучшить качество визуализации результатов встраивания данных из исходного метрического пространства в метрическое пространство меньшей размерности.

Результаты расчетов

Апробация предлагаемого подхода была выполнена на группе из 89 коротких ВР социально-экономической сферы Рязанской области, длина которых была ограничена 5 отсчётами времени [7].

При этом t-SNE- и UMAP-алгоритмы при значениях параметров, заданных по умолчанию, были применены к 3 группам коротких ВР социально-экономической сферы Рязанской области, длина которых была равна 3, 4 и 5 отсчётам времени, с целью сравнительного анализа в контексте выявления оптимального числа кластеров, а также возможного изменения кластерной принадлежности отдельных ВР при увеличении их длины на единицу (с 3-х до 4-х и с 4-хдо 5-ти). При этом предварительно для каждой группы ВР была выполнена нормировка в [0, 1]. Результаты визуализации группы ВР с применением t-SNE- и UMAP-алгоритмов приведены на рисунке 1.

При этом время расчетов с применением t-SNE- и UMAP-алгоритмов при 3, 4 и 5 отсчётах времени составило соответственно 2.127 с и 0.317 с; 1.712 и 0.285; 2.236 и 0.285. Очевидно, что UMAP-алгоритм обладает большим быстродействием, чем t-SNE-алгоритм: время при работе с UMAP-алгоритмом при 3, 4 и 5 отсчётах времени оказалось меньше соответственно в 6.710, 6.007 и 7.846 раз.

Результаты расчетов позволяют сделать следующие выводы: при значениях параметров, заданных по умолчанию, UMAP-алгоритм в большинстве случаев позволяют расположить объекты разных кластеров на большем расстоянии, чем это делает t-SNE-алгоритм. При этом UMAP-алгоритм при визуализации позволяет предположить, что число кластеров равно 3 при числе отсчётов, равном 3, в то время как при числе отсчётов, равном 4 и 5, число кластеров можно положить равным 2. t-SNE-алгоритм во всех трех случаях визуально позволяет предположить существование 2 кластеров (хотя при числе отсчётов, равном 3, целесообразно рассмотреть случай разбиения группы ВР на 3 кластера), однако при числе отсчетов, равном 4 и 5, кластеры отделимы хуже, чем в случае работы с UMAP-алгоритмом. Очевидно, что в дальнейшем можно ограничиться анализом результатов кластеризации только для случая двух и трех кластеров. Анализ номеров объектов (ВР), входящих в полученные «сгустки» (кластеры) в двухмерном пространстве, может позволить сделать предположение о перемещении отдельных ВР из одних кластеров в другие (и, следовательно, сделать предположение о проявлении структурных трансформаций в группе ВР с течением времени).

В качестве потенциально целесообразных к применению алгоритмов кластеризации для рассматриваемой групп ВР следует назвать алгоритм иерархической агломеративной кластеризации [7], а также DBSCAN-алгоритм (Density-based spatial clustering of applications with noise) [12], реализующий кластеризацию объектов с учетом плотности: алгоритм группирует вместе объекты, которые близко расположены (т.е. объекты со многими ближайшими соседями), помечая как выбросы объекты, которые являются единичными в областях с малой плотностью (т.е. объекты, ближайшие соседи которых лежат далеко).

Эффективность выбранных алгоритмов кластеризации была подтверждена экспериментально. При этом алгоритм иерархической агломеративной кластеризации при использовании евклидовой метрики и метода Уорда определил число 2 в качестве оптимального чис-

ла кластеров при 3, 4 и 5 отсчётах времени (на основе значений индекса кластерного силуэта и показателя локтя (колена). DBSCAN-алгоритм в случае 3 и 4 отсчётов времени выделил 2 кластера, а в случае 5 отсчётов времени – 3 кластера. При этом число шумовых объектов при 3 4 и 5 отсчётах времени оказалось соответственно равным 9, 0 и 0, а значение индекса кластерного силуэта [13] – 0.389, 0.473 и 0.619 соответственно.

Следует отметить, что, например, алгоритм нечетких *c*-средних (FCM-алгоритм) оказался менее эффективным при решении задачи кластеризации рассматриваемой группы ВР, что может быть объяснено, в том числе, результатами визуализации (рисунок 1): очевидно, что форма кластеров в исходном метрическом пространстве в большинстве случаев далека от гиперсферической.

Выводы

Предлагаемый подход к визуализации данных позволяет выдвинуть предположение об оптимальном числе кластеров и составить список алгоритмов кластеризации, которые целесообразно применить к анализируемому набору данных. При этом следует выполнить исследование результатов визуализации при разных значениях параметров t-SNE- и UMAP-алгоритмов.

Библиографический список

1. Jain A.K., Dubes R.C. Algorithms for Clustering Data. Prentice Hall. 1988. 334 p.
2. Milligan G.W., Cooper M.C. An Examination of Procedures for Determining the Number of Clusters in a Data Set // Psychometrika. 1985. Vol. 50. pp. 159–179.
3. Peter H.A., Sneath R.S. Numerical Taxonomy. The principles and practice of numerical classification. Francisco: W.H. Freeman and Co. 1973. 573 p.
4. Bezdek J.C., Ehrlich R., Full W. FCM: The fuzzy c-means clustering algorithm // Computers & Geosciences. 1984. Vol. 10 (2-3). pp. 191 − 203.
5. Демидова Л.А., Титов С.Б. Подход к проблеме нечеткой кластеризации в условиях неопределенности выбора целевой функции // Вестник Рязанского государственного радиотехнического университета. 2009. № 29. С. 54 – 60.
6. Stepanov M.A., Demidova L.A. Algorithms of Identifying of the Structural Transformations in Time Series Groups // 8th Mediterranean Conference on Embedded Computing, MECO 2019. 2019. pp. 234 – 237.
7. Степанов М.А. Подход к решению задачи выявления структурных трансформаций в группах временных рядов // Cloud of Science. 2019. Т. 6 № 2. С. 201–226.
8. Степанов М.А. Методика выявления структурных трансформаций временных рядов с использованием принципов нечеткой кластеризации // Вестник Рязанского государственного радиотехнического университета. 2019. № 69. С. 149 – 159.
9. Astakhova N.N., Demidova L.A., Nikulchev E.V. Forecasting of Time Series' Groups with Application of Fuzzy C-Mean Algorithm // Contemporary Engineering Sciences. 2015. Vol. 8. No. 35. pp. 1659 – 1677.
10. van der Maaten L.J.P., Hinton G.E. Visualizing Data Using t-SNE // Journal of Machine Learning Research. 2008. Vol. 9.
11. McInnes L., Healy J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, ArXiv e-prints 1802.03426. 2018.
12. Ester M., Kriegel H.-P., Sander J., Xu X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise // KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. 1996. pp. 226 – 231.
13. Rousseeuw P. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis // Journal of Computational and Applied Mathematics. 1987. Vol. 20(1). pp. 53–65.

УДК 004.41; ГРНТИ 50.05.13

ПОВЫШЕНИЕ УРОВНЯ АБСТРАКЦИИ РАЗРАБОТКИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ С ИСПОЛЬЗОВАНИЕМ ВИЗУАЛЬНОГО МЕТОДА

А.И. Николаев

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, andrey.nikolaev130@gmail.com

Аннотация. В работе рассматриваются возможности повышения уровня абстракции, с которой работает программист при разработке параллельных программ. Рассмотрены примеры реализации, которые встречаются при использовании потоков в высокоуровневых языках программирования. Рассматривается возможность использования визуального метода с целью повышения уровня абстракции.

Ключевые слова: параллельное программирование, уровень абстракции, визуальный метод, поток, визуализация, мгогопоточность.

INCREASING THE LEVEL OF ABSTRACTION OF THE DEVELOPMENT OF PARALLEL PROGRAMS USING THE VISUAL METHOD

A.I. Nikolaev

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, andrey.nikolaev130@gmail.com

The summary. The paper considers the possibilities of increasing the level of abstraction with which the programmer works when developing parallel programs. Implementation examples that are encountered when using streams in high-level programming languages are considered. The possibility of using the visual method to increase the level of abstraction is considered.

Keywords: parallel programming, level of abstraction, visual method, thread, visualization, multithreading.

Использование параллельных программ ускоряет работу различных алгоритмов путем распараллеливания вычислений. Написание параллельных программ довольно сложная задача, особенно для разработчиков, которые только начитают свое знакомство с разработкой параллельных алгоритмов [1]. При написании параллельных программ требуется хорошее владение механизмами языка, в противном случае возрастает риск ошибки в алгоритме, и найти и устранить неточность, в случае многопоточных вычислений достаточно тяжело, а иногда невозможно без использования дополнительных средств [2]. Так же возникает проблема в скорости работы, если хоть один поток будет работать крайне медленно, то все остальные будут его ожидать и это приведет к тому, что наша программа будет похожа на последовательное выполнение команд, как в случае с однопоточной системой [3].

Для упрощения написания параллельных программ в высокоуровневых языках вводят различные уровни абстракции, так как детали реализации могут быть крайне сложные для восприятия человеком. Так на самом нижнем уровне машина оперирует нулями и единицами. Так как это не совсем удобно и легко воспринимаемо человеком, был придуман язык ассемблера, что является символьным представлением двоичного языка компьютера. С течением времени появились более высокоуровневые языки программирования, такие как Java, C# и т.п. В этих языках программист пишет текст программ с использованием операторов языка. Разработчику уже нет необходимости править двоичные коды программ, все это возлагается на компиляторы и трансляторы, которые переводят текст программ на язык понятный машине. Если рассмотреть подробнее устройство Java, то в ней присутствует виртуальная машина, которая реализована для различных операционных систем, это позволяет запускать одни и те же программы в абсолютно разных средах. За счет всего этого повышается уровень абстракции, с которой работает программист.

Абстракция в системах обычно делится на несколько уровней и на каждом уровне абстракция должна быть согласована. Таким образом вся система разделяется на несколько

уровней, которые могут работать с другими уровнями относительно автономно. И так же программисту нет необходимости держать в голове информацию об абстракциях других уровней. Абстракции помогают в борьбе со сложностью систем. Они так же уменьшают связанность разрабатываемых программ и обеспечивают взаимозаменяемость различных компонентов.

При разработке параллельных программ с использованием высокоуровневого языка, программист не работает напрямую с потоками в системе. Так, например, в языке программирования Java есть свои классы и методы для работы с потоками, программисту нет необходимости напрямую взаимодействовать с потоком, это оставлено для уже существующих в языке средств. Пример подобных абстракций – класс Thread, у которого есть набор необходимых методов для работы с потоком. Программист может работать с высокоуровневой абстракцией, не опускаясь в детали, это позволяет снизить сложность разрабатываемых алгоритмов, а также уменьшить количества ошибок допущенных при разработке программных систем с использованием параллелизма. При необходимости опытный разработчик спуститься на несколько уровней абстракции ниже, и сделать более точные настройки, но это необходимо в исключительных ситуациях. При обычном использовании механизмов много поточности для построения параллельных программ достаточно использования высокоуровневой абстракции, которую предоставляет язык программирования.

Использование возможностью управления потоков для написания параллельных программ на низком уровне увеличивает время необходимую на разработку и отладку новых алгоритмов, при этом ухудшается качество, так как велика вероятность ошибки. Так же очень тяжело разбираться новому программисту в уже написанном коде и это увеличивает сложность поддержки и обслуживания существующего кода. Помимо увеличения времени и сил на разработку, так же повышается уровень входа для начинающих разработчиков, что так же негативно влияет на данное направление. Для написания эффективных параллельных алгоритмов требуются более опытные разработчики, которые стоят также дороже и их количество обычно ограничено. Так же увеличиваются затраты на подготовку новых специалистов. Для обучения требуется больше времени и так же требования к квалификации преподавателей тоже растет. Гораздо эффективнее работать с высокоуровневыми абстракциями и не нарушать работу на низком уровне.

Рассмотрим и сравним существующие механизмы, которые существуют в языка Java и C# для написания параллельных программ и работы с потоками. В языке программирования Java главным для работы с потоками является класс Thread [4]. Он имеет в себе следующий методы:

- getName() - получить наименование потока;
- getPriority() - получить приоритет потока;
- isAlive() - определить, находится ли поток в состоянии выполнения;
- join() – приостановить выполнение текущего потока и ожидать окончание выполнения потока;
- run() - внутри этого метода пишется текст программы, который должен выполняться в отдельном потоке;
- sleep() – поставить поток на паузу на указанное время;
- start() - запустить выполнение потока.

Разработчик оперирует данными методами и не спускается на абстракцию ниже для того, чтобы посмотреть, как это работает, ему нет необходимости это знать. Но для большинства методов синхронизации, таких как семафор, флаг, мьютекс, критическая секция, событие, замок с обратным отсчетом, обменник достаточно стандартных механизмов языка [5].

В языке программирования C# так же существует класс Thread и у него есть методы и свойства, которые являются эквивалентами методов в Java:

- Name – получить имя потока, который является его идентификатором;
- Priority – получить приоритет потока;
- IsAlive() - определить, находится ли поток в состоянии выполнения;
- Join() - приостановить выполнение текущего потока и ожидать окончание выполнения потока;
- Sleep() – поставить поток на паузу на указанное время;
- Start() - запустить выполнение потока.

Отличием является метод run() в случае с Java текст программы, который необходимо выполнить пишется в этом методе. В C# текст программы, который необходимо выполнить передается в конструкторе при создании. Механизмы синхронизации можно описывать как и в Java с помощью стандартных операторов языка.

В большинстве высокоуровневых языков существуют абстракции для работы с потоками, это позволяет использовать визуальный метод создания параллельных программ совместно с кодогенерацией. Мы можем один раз нарисовать необходимое нам поведение и сформировать код на конкретном языке программирования за счет модулей преобразования внутреннего представления визуального отображения. Так как во многих языках существуют похожие конструкции для работы с потоками этот процесс формирования кода будет возможен для различных языков. В случае сильного отличия придется просто доработать соответствующий модуль преобразования. За счет того, что у визуального представления имеется единое внутреннее интерпретация с определенной структурой, эту структуру можно проверять на соответствие правилам формирования параллельных программ, что позволит избежать ошибок еще до этапа генерации текста программы. Необходимо установить четкие правила по которым внутреннее представление будет преобразовываться в ту или иную конструкцию языка и эти правила должны быть эквивалентными для всех языков, для которых поддерживается кодогенерация.

Использование разработчиком кодогенерации из внутреннего представления позволяет оперировать различными представлениями предметной области или задачи. Так как внутреннее представление помимо текста программы может быть отображено в каком-либо удобочитаемом и удобно редактируемом формате. Одним из таких представлений является визуальное представление. В этом случае параллельный алгоритм строится в визуальную картинку, с которой работает программист. Это позволяет повысить уровень абстракции и сосредоточится на решении поставленной задачи и не отвлекаться на детали реализации. Так же этот подход может иметь положительное воздействие при обучении. Человеку, которые впервые сталкивается с понятиями поток и параллельные алгоритмы достаточно тяжело это понять с первого раза. Гораздо проще воспринимать подобную информацию с использованием визуального представления. Это позволит дать студентам в вузе базовое представление о работе параллельных программ за более короткий срок. При разработке крупных систем, визуализацию можно использовать при необходимости разработки каких либо параллельных алгоритмов с высокой долей абстракции. При этом будет сгенерирован текст программы в визе каркаса, которые останется наполнить бизнес логикой. Все необходимые синхронизации между потоками уже будут настроены. Это позволит сильно сократить время разработки и повысить качество. Кроме того это позволит представить сложные алгоритмы в более наглядном виде, что может способствовать повышению качества передачи знаний. При разработке крупных систем разработчики и архитекторы могут меняться и необходимо передавать все полученные знания своим преемникам, для этого служит документация. Визуальный метод является отчасти самодокументирумым, так как представляется в наглядном виде.

Кроме рассмотренных преимуществ в использовании абстракции высокого уровня есть некоторые недостатки. При необходимости внести изменение на каком либо из уровней абстракции, эти изменения приходится вносить во все остальные промежуточные уровни.

Так же использование абстракции от источников данных может дать неоптимальный вариант использования этого источника, не учитывается специфика работы с необходимым источником, кроме того иногда достаточно неочевидно, как будет строится текст программы по визуальному представлению и достаточно тяжело с этим разбираться. Частично эти проблемы возможно решить добавлением большего количества метаинформации для модели при визуальном проектировании параллельных программ.

Данный подход рассматривается как продолжение исследований развивающихся методов программирования [6-7].

Библиографический список

1. Богачев К .Ю., Основы параллельного программирования. – М.: БИНОМ. Лаборатория знаний, 2014 – 342 с.

2. McConnell S. C. Code complete, 2nd Edition. – Redmond, Washington: Microsoft Press, 2004 – 914 p.

3. Николаев А.И. Разработка визуального метода проектирования параллельных программ // Новые информационные технологии в научных исследованиях – НИТ-2018: материалы XXII Всероссийск. научн.-техн. конф. студентов, молодых ученых и специалистов./Рязан.гос.радиотехн.ун-т. – Рязань, 2018. – С. 182-183.

4. Oaks S., Wond H. Java Threads, Third Edition. – Sebastopol: O'Reilly Media, 2009 – 360 p.

5. Николаев А.И. Визуализация параллельного программирования // современные технологии в науке и образовании-2019: материалы II международного научнотехнического форума том 4./ Рязан.гос.радиотехн.ун-т. – Рязань, 2019. – с. 180-183.

6. Пруцков А.В., Цыбулько Д.М. Применение проблемно-ориентированного объектного программирования для описания порядка работы интеллектуальных и информационных систем // Вестник Рязанского государственного радиотехнического университета. – 2014.– № 47. – С. 92-96.

7. Пруцков А.В., Цыбулько Д.М. Проблемно-ориентированный подход к пользовательскому программированию // Cloud of Science. – 2016. – Т. 3. – № 1. – С. 105-114.

УДК 004; ГРНТИ 20.01.45

АВТОМАТИЗАЦИЯ МОРФОЛОГИЧЕСКОГО АНАЛИЗА В ПРОМЫШЛЕННЫХ СИСТЕМАХ ОБРАБОТКИ ТЕКСТОВ

Н.Ш. Мадибрагимов

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, navruzmadibragimov@gmail.com

Аннотация. В статье описывается автоматическая обработка текстов, рассматриваются промышленные системы обработки текстов, разработанные на сегодняшний день. Также приводятся методы морфологического анализа на вычислительных машинах. Объясняется метод генерации и определения форм слов и его реализация для таджикского языка.

Ключевые слова: автоматическая обработка текстов, автоматический морфологический анализ, автоматическая обработка таджикского языка.

AUTOMATION OF MORPHOLOGICAL ANALYSIS IN INDUSTRIAL TEXT PROCESSING SYSTEMS

N.Sh. Madibragimov

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, navruzmadibragimov@gmail.com

The summary. The article describes automatic word processing, considers industrial word processing systems developed to date. Methods of morphological analysis on computers are also given. The method of generating and determining the forms of words and its implementation for the Tajik language are explained.

Keywords: automatic text processing, automatic morphological analysis, automatic Tajik language processing.

Автоматическая обработка текстов

Необходимость и важность автоматической обработки текстов (АОТ), обусловлена с расширением доступности интернета по всему миру и увеличением текстовых данных в сети. Для преобразования текста на естественном языка на язык компьютера, разработаны множество прикладных систем, представляющих коммерческий интерес, такие как машинный перевод, информационный поиск, генерация текста и т. д. Теоретическую основу АОТ составляет компьютерная лингвистика, использующая методы машинного обучения, статистического анализа, модели Маркова, логические модели и модификации этих методов для обработки «больших данных» [1]. Существует несколько подходов к такой модификации: распараллеливание алгоритмов, применение методов снижения размерности, предварительная обработка данных, в ходе которой целостные тексты заменяются на их отдельные подтексты [2]. Лингвистические методы могут быть универсальными для многих языков, несмотря на различие между национальными языками. Некоторые морфологические и синтаксические модели удается использовать для анализа текстов как на русском, так и на таджикском языке.

Промышленные системы обработки текстов

Сфера применения программных приложений компьютерной лингвистики постоянно расширяется. К самым популярным проблемам компьютерной лингвистики, относится машинный перевод, к которому относятся системы от больших интернациональных исследовательских проектов до коммерческих автоматических переводчиков. Существует направление машинного перевода – статистическая трансляция, опирающаяся на статистику переводных пар слов и словосочетаний, развитие которой связан с машинным обучением и нейронными сетями, исследуемые в рамках искусственного интеллекта [3].

Также популярной и важной задачей компьютерной лингвистики является информационный поиск (Information Retrieval) и связанные с ней задачи реферирования, аннотирование, классификация, кластеризация, рубрицирования документов.

Для правильного нахождения текстов из больших текстовых баз данных, необходимо произвести индексирование текстов, которое требует лингвистической предварительной обработки текстов и создание специальных индексных структур.

Сегодня развивается еще одно направление распознавание и синтез звучащей речи, для которой также применяется машинное обучение и ошибки исправляются автоматическим методами с помощью словарей.

Методы морфологического анализа

Основными задачами морфологического анализа являются:

- выделение из текста словоформ;
- распознавание слов или словосочетаний;
- нормализация словоформ, т.е. приведение слова к словарному виду (лемматизация);
- распознавание грамматических признаков словоформ, такие как часть речи, падеж, число и т.п. [4].

Существуют два основных подхода для решения задач. Морфологический анализ со словарем и морфологический анализ без словаря. При первом подходе словари дадут максимальную информацию по форме известного слова, если текст написан без опечаток. Второй подход, метод без словаря используют алгоритмы, которые умеют преобразовать слова в различные грамматические формы. Такие алгоритмы можно разделить на вероятностно-статистические методы и лексиконы основ и суффиксов [4].

«Любой метод морфологического анализа и/или синтеза включает две части: декларативную, которая состоит из данных, структурированных в словарях и таблицах, необходимых для морфологического анализа и синтеза; процедурной – включающей алгоритмы морфологического анализа и синтеза, вспомогательные процедуры» [5].

Метод генерации и определения форм слов

При морфологической обработке текстов, решаются задачи генерации и определения форм слов. Генерация формы слова, т.е. продукция словоформ – это процесс получения формы с использованием в качестве начальных параметров, основы и грамматического значения. Определение формы слова – это процесс, обратный генерации, иначе говоря нахождение по данной словоформе ее основу и грамматического значения.

Универсальный метод генерации и определения форм слов должен отвечать следующим критериям:

- универсальность генерации и определения форм слов естественных языков различных групп и семейств;
- универсальность структуры словарей, не требующей конвертации для решения задач определения или генерации;
- универсальность метода для всех видов формообразования словоформ, обработки всей парадигмы слова, в том числе и аналитических словоформ.

Существует модель формообразования и метод генерации и определения форм слов, который удовлетворяет перечисленным критериям и включает также алгоритмы генерации и определения, разработанные на основе этой модели.

«Модель формообразования естественного языка заключается в том, что получение любой словоформы с данным грамматическим значением можно представить в виде последовательности конечного числа преобразований над основой» [9].

Реализация метода генерации и определения форм слов таджикского языка

В настоящее время ведется работа по классификации изменения слов и принципов словообразования в таджикском языке, для их описания с помощью метода и генерации и определения форм слов [6]. В работе [7] исследованы префиксы и постфиксы частей речи таджикского языка, и выявленные аффиксы приведены в таблицах соответствующих частей речи. Для существительного показаны 1801 выявленные аффиксы.

На основе этих данных, исследуются 1381 существительных таджикского языка из словаря книги «Таджикский язык» С. Арзуманов, А. Сангинов [8]. На момент написания этих тезисов доклада определены следующие типы формообразования:

1. Тип 1.1. Основы слов, заканчивающиеся на согласные буквы (*б, в, г, ғ, д, ж, з, к, қ, л, м, н, п, р, с, т, ф, х, ҳ, ч, ҷ, ш*), и которым в зависимости от окончания и от стиля, свойственно образование множественного числа только при помощи постфикса ***-ҳо***. Этот тип характеризуется тем, что при морфологическом преобразовании сами слова не меняются, а постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, при добавлении которых, существительные в предложении выступают как сказуемое, применяются из 1-вида постфиксов (*+ат, +аш* и т.д.). Пример: *телефон – телефонам, телефонат, телефонаш, …, телефони, телефонро…, телефонҳо…*

2. Тип 1.2. Основы слов, заканчивающиеся на согласные буквы (*б, в, г, ғ, д, ж, з, к, қ, л, м, н, п, р, с, т, ф, х, ҳ, ч, ҷ, ш*), и которым в зависимости от окончания и от стиля, свойственно образование множественного числа также при помощи постфикса ***-он***. При морфологическом преобразовании сами слова не меняются, а постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Слова принимают постфикс *-он*, для образования множественного числа. Предикативные связки, при добавлении которых, суще-

ствительные в предложении выступают как сказуемое, применяются из 1-вида постфиксов (*+ат, +аш* и т.д.). Пример: *сухан – суханам, суханат, суханаш, ..., сухани, суханро, суханҳо, суханон...*

3. Тип 2.1. Основы слов, заканчивающиеся на гласные буквы (*а, е, ё, и, о, у, ӯ, э, ю, я*), и которым в зависимости от окончания и от стиля, свойственно образование множественного числа только при помощи постфикса ***-ҳо***. Постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, применяются из 2-вида (*+ятон, +яшон* и т.д.). Пример: *ручка – ручкаям, ручкаят, ручкаятон, ручкаяшон, ..., ручкаи, ручкаро, ручкаҳо…*

4. Тип 2.2. Основы слов, заканчивающиеся на гласные буквы «*а», «я»*, и которым в зависимости от окончания и от стиля, свойственно образование множественного числа также при помощи постфикса ***-гон***. Постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, применяются из 2-вида (*+ятон, +яшон* и т.д.). Пример: *нависанда – нависандаям, ..., нависандаяшон, нависанда ҳо, нависандагон; ҳамсоя – ҳамсояям, ҳамсоят, ..., ҳамсояҳо, ҳамсоягон…*

5. Тип 2.3. Основы слов, заканчивающиеся на гласные буквы *«о»*, и которым в зависимости от окончания и от стиля, свойственно образование множественного числа также при помощи постфикса ***-ён***. Постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, применяются из 2-вида (*+ятон, +яшон* и т.д.). Пример: *бобо – бобоям, ..., бобояшон, бобоҳо, бобоён…*

6. Тип 2.4. Основы слов, заканчивающиеся на гласные буквы *«у», «ӯ»*, и которым в зависимости от окончания и от стиля, свойственно образование множественного числа также при помощи постфикса ***-вон***. Постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, применяются из 2-вида (*+ятон, +яшон* и т.д.). Пример: *оҳу – оҳуям, ..., оҳуяшон, оҳуҳо, оҳувон; абрӯ – абрӯям, ..., абрӯяшон, абрӯҳо, абрӯвон…*

7. Тип 3.1. Основы слов, заканчивающиеся на «*й»* (*и-и кӯтоҳ*), и множественное число образуется только при помощи постфикса ***-ҳо***. При добавлении предикативных связок, буква «*й*», в конце удаляется и применяются из 2-вида. Остальные постфиксы и вспомогательные слова добавляются по общим правилам. Пример: *най – найи, найро, наям, наят, наяшон, найҳо...*

8. Тип 3.2. Основы слов, заканчивающиеся на «*й»* (*и-и кӯтоҳ*), и множественного число образуется также при помощи постфикса ***-ён***. При добавлении предикативных связок, буква «*й»*, в конце удаляется и применяются из 2-вида. Остальные постфиксы и вспомогательные слова добавляются по общим правилам. Пример: *бой – бойи, бойро, боям, боят, бояшон, бойҳо, боён…*

9. Тип 4.1. Основы слов, заканчивающиеся на безгласный «ъ» и буква, стоящая перед «ъ» - согласная. Постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, применяются по принципу Типа 1.1. Пример: *навъ – навъам, навъаш, ... навъро...*

10. Тип 4.2. Основы слов, заканчивающиеся на безгласный «ъ» и буква, стоящая перед «ъ» - гласная. Постфиксы и вспомогательные слова добавляются по общим правилам, не изменяя основу. Предикативные связки, применяются по принципу Типа 2.1. Пример: *мавзӯъ – мавзӯъям, мавзӯъяш, ... мавзӯъро…*

11. Тип 5.1. Основы слов, заканчивающиеся на букву «*ӣ»* (*и-и заданок*), и множественное число образуется только при помощи постфикса ***-ҳо***. При добавлении постфикса, буква «*ӣ»* в конце заменяется на «*и»*. Предикативные связки, как и слова с гласным окончанием, применяются из 2-вида. Вспомогательные слова добавляются по общим правилам. Пример: *сабзӣ – сабзии, сабзиро, сабзиям, сабзият, сабзиҳо…*

12. Тип 5.2. Основы слов, заканчивающиеся на букву «*ӣ*» (*и-и заданок*), и множественное число образуется при помощи постфиксов ***-ҳо, -ён***. При добавлении постфикса, буква «*ӣ*» в конце заменяется на «*и*». Предикативные связки, как и слова с гласным окончанием, применяются из 2-вида. Вспомогательные слова добавляются по общим правилам. Пример: *мураббӣ – мураббии, мураббиро, мураббиям, мураббият, мураббиҳо*, *мураббиён…*

Предикативные связки, постфиксы, обозначающие притяжательные местоимения, бывают 2 вида:

- 1-вид – постфиксы *+ам, +ат, +аш, +амон, +ашон, +атон, +анд,* применяются к существительным с согласным окончанием;

- 2-вид *+ям, +ят, +яш, +ямон, +яшон, +ятон, +янд* используются для существительных с гласным окончанием.

Закючение

В результате исследования были определены 12 типов формообразования слов. Генерация словоформ таджикского языка осуществлен на языке программирования PHP7 в виде веб-сайта, с помощью фреймворка CodeIgniter 3.1.10, на основе знаний авторов о морфологии таджикского языка и на основе консультаций экспертов. Практическая реализация тестового проекта находится на локальном сервере. Для внешнего интерфейса использован простой и легко настраиваемый HTML, CSS, и Javascript инструментарий, фреймворк Bootstrap 4.0 от Twitter.

Словарь основ сгенерирован из книги «Таджикский язык» С.Д. Арзуманова с добавлением нескольких слов от автора, база постфиксов используется из диссертации Г.М. Довудова «Компьютерный морфологический анализ таджикских словоформ». Все полученные результаты генерации словоформ проверены и проанализированы экспертными знаниями специалистов по морфологии таджикского языка. В настоящее время ведется работа по исследованию слов прилагательных и глаголов таджикского языка.

Исследованные существительные таджикского языка и их словоформы, будут использованы при классификации слов в программной реализации универсального метода генерации и определения форм слов.

Библиографический список

13. Kuznetsov, S.O. Fitting Pattern Structures to Knowledge Discovery in Big Data [Текст] / S.O. Kuznetsov. – Moscow: ICFCA, 2013. PP. 254-266.

14. Ильвовский, Д., Черняк, Е. Системы автоматической обработки текстов [Электронный ресурс] / Д. Ильвовский, Е. Черняк // Открытые системы. СУБД, 2014, № 1. URL: https://www.osp.ru/os/2014/01/13039687/

15. Большакова, Е.И., Воронцов, К.В., Ефремова, Н.Э., Клышинский, Э.С., Лукашевич, Н.В., Сапин, А.С. Автоматическая обработка текстов на естественном языке и анализ данных: учебное пособие. – М.: НИУ ВШЭ, 2017. – 269 с.

16. Кириллов, М. Text Mining: морфологический анализ [Электронный ресурс] // М. Кириллов. – 2016 г. URL: https://morphs.ru/posts/2016/08/18/text-mining-morph.

17. Пруцков, А.В., Розанов, А.К. Методы морфологической обработки текстов [Текст] // Прикаспийский журнал: управление и высокие технологии. 2014. № 3 (27). С. 119-133.

18. Мадибрагимов, Н. Компьютерные модели формообразования слов и их применение для описания морфологии таджикского языка // Современные технологии в науке и образовании – СТНО-2018 [Текст]: сб. тр. междунар. науч.-техн. и науч.-метод. конф.: в 10 т. Т.1. / под общ. ред. О.В. Миловзорова. – Рязань: Рязан. гос. радиотехн. ун-т, 2018; Рязань. – С. 65-68.

19. Довудов, Г.М. Компьютерный морфологический анализ таджикских словоформ. [Текст]: дисс…..канд. техн. наук: 05.13.11: защищена 06.04.2018/ Довудов Гулшан Мирбахоевич. – Душанбе, 2018. – 161 с.

20. Арзуманов, С.Д., Сангинов, А. Таджикский язык. – Душанбе: Маориф, 1988 – 416 с.

9. Пруцков, А.В., Модели, методы и программы автоматической обработки форм слов в естественно-языковых интерфейсах [Текст] дисс…..док. техн. наук: 05.13.11: защищена 23.12.2015/ Пруцкова Александра Викторович. – Рязань, 2015. – 279 с.

УДК 004.42; ГРНТИ 50.05.19

РЕАЛИЗАЦИЯ ПЕРВИЧНОЙ ОЦЕНКИ СВОЙСТВ ВРЕМЕННЫХ РЯДОВ

Е.А. Попова

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, aaa_rrr@mail.ru

Аннотация. В работе рассматривается алгоритм первичной оценки статистических зхарактеристик временного ряда для принятия решения о целесообразности дальнейшей работы с ним.

Ключевые слова: временной ряд, загрузка, построение графика, гитограмма, сезонность, тренд, цикл, фрактальность, долговременная память.

TIME SERIES PROPERTIES ASSESSMENT IMPLEMENTATION

E.A. Popova

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, aaa_rrr@mail.ru

The summary. The paper discusses the implementation of time series assessment to make a decision wether is possible to make a model and forecast future values.

Keywords: time series, loading plotting, histogram, seasonal, trend, cycle, fractal, long memory.

Временной ряд (ВР): $y_1, y_2, ..., y_T$, где $y_t \in R$ – значения признака, измеренные через постоянные временные интервалы [1].

Опишем типичный сценарий обработки временного ряда и его реализацию с помощью языка программирования Python и библиотеки pandas.

Для представления временного ряда в памяти воспользуемся объектом dataframe из библиотеки pandas. После импорта сырых данных из источника осуществляется проверка, присутствия пустых или неопределенных значений, переименование столбцов, если необходимо, конвертация значения дат к типу datetime и установка индекс для дальнейшей работы.

В качестве примера рассматривается временной ряд, представляющий собой частоту поиска слова «vacation» в запросах поисковой системы Google (https://trends.google.com/trends/explore?date=all&geo=US&q=vaction) в период с января 2004 по октябрь 2019.

Необходимо оценить основные статистические характеристики ряда, построить его график, посмотреть частоту распределений его значений с помощью гистограммы и произвести ядерную оценку плотности. Оценка характеристик ряда, полученная с помощью функции describe показана в таблице 1. График ряда представлен на рисунке 1.

Таблица 1. Оценка статистических характеристик временного ряда

Характеристика	Значение
Count	193.00
Mean	40.56
Std	20.49
Min	17.00
Max	100.00

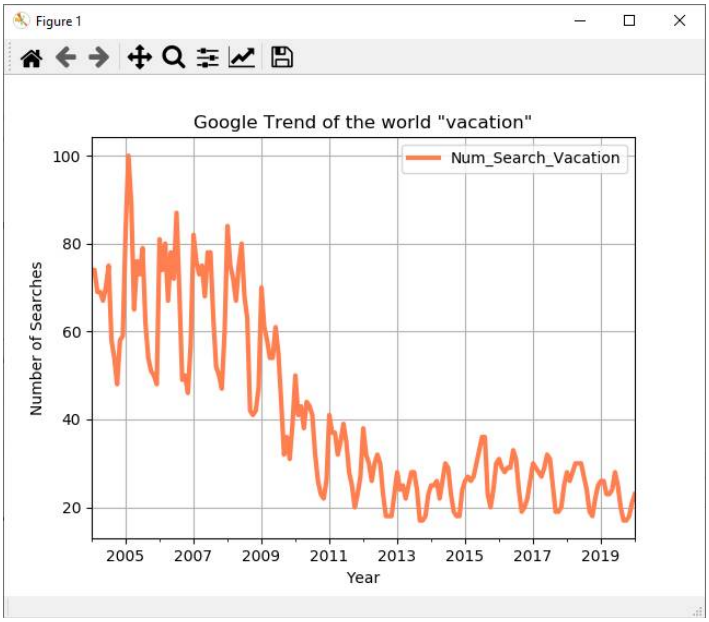


Рис. 1. График временного ряда

Визуальная оценка временного ряда выявляет трендовое снижение и последующую стабилизацию в районе 2013 года. Также периодические или циклические изменения, особенно в октябре каждого года и значительный всплеск в июне 2015 года. Линовка графика помогает определять периодические изменения, которые повторяются каждый год.

Для оценки распределения частоты появления значений в ряду строится гистограмма распределения значений (рисунок 2).

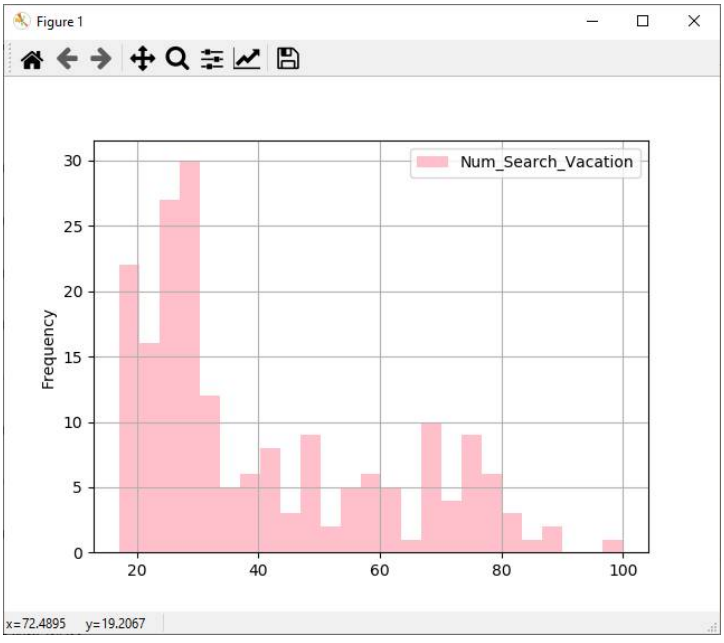


Рис. 2. Гистограмма распределения значений

Для визуальной оценки распределения временного ряда строится график ядерной оценки плотности. График сглаживает шум, не зависит от количества столбцов, в отличие от гистограммы. Присутствующий в ряду, пик указывает на преобладающие значения (рисунок 3).

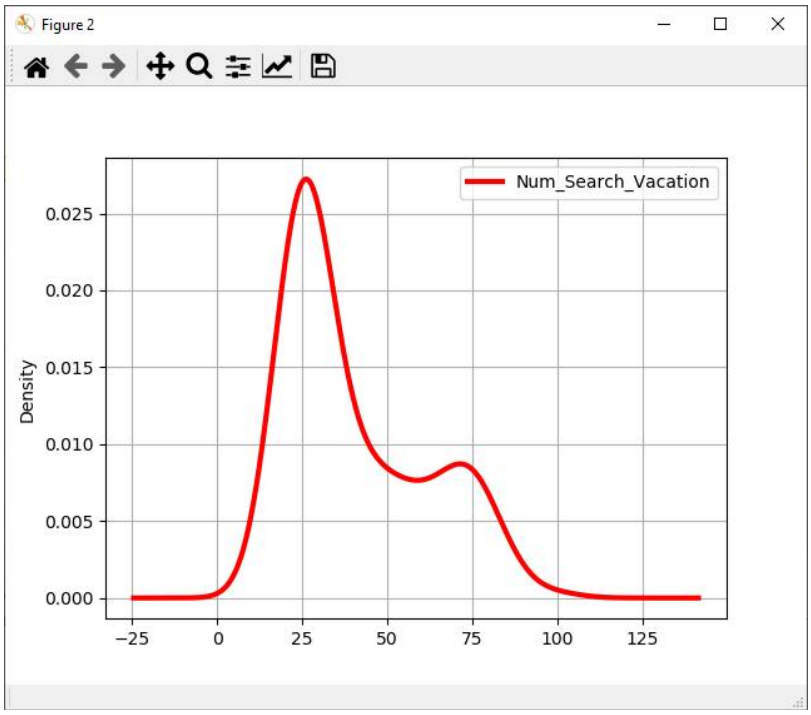


Рис. 3. График ядерной оценки плотности

Далее производится декомпозиция временного ряда, т.е. выделение тренда, сезонности, цикла и случайной составляющей (нерегулярности) [2].

- тренд, представляющий собой общую систематическую линейную или нелинейную компоненту, которая может изменяться во времени;- структурировать объекты по типовым решениям, по сферам деятельности, по типу операции, которую разработка автоматизирует (например, учет больничных, расчет отпускных и т.д.);

- сезонность, представляющая собой периодически повторяющуюсякомпоненту (периоды сезонов обычно фиксированы и известны);

- циклы, представляющие собой долговременные колебания с нефиксированным периодом;

- нерегулярности, представляющие собой непредсказуемые случайные значения, называемые белым шумом.

Декомпозиция может быть записана с использованием множества форм. В частности, существуют аддитивная и мультипликативная формы. Выбор уравнения для записи декомпозиции зависит от модели, которая будет использована в дальнейшем.

График компонент временного ряда (рисунок 4) подтверждает те выводы, которые были селаны на основе первичной оценки ряда. Перед построением модели следует избавиться от влияния составляющих ряда, например, выполнив дифференцирование.

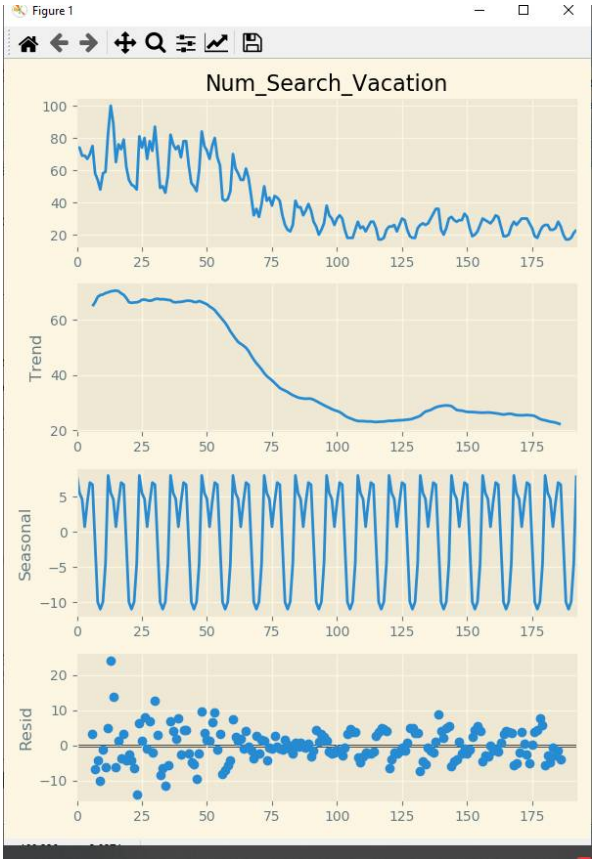


Рис. 4. Компоненты временного ряда

Кроме встроенных функций библиотеки pandas представляется интересным проверить, фрактальные характеристики временного ряда.

Фрактальная размерность, является показателем сложности кривой, ее неровности. По величине фрактальной размерности, исследуя ее изменения и чередования, можно предсказывать поведение системы и выявить нестабильные состояния, т.е. прогнозировать и классифицировать

Для этого можно оценить показатель Херста, который является показателем фрактальноси ряда и может принимать следующие значения [3]:

- $H = 1$ – гладкая кривая, процесс предсказуем;
- $H > 0.5$ – персистентный процесс (черный шум), инерционный, имеющий долговременную память, имеет склонность следовать трендам; при этом временной ряд близок по структуре к фракталу с положительной реакцией и наблюдается склонность следовать трендам и циклам;
- $H = 0.5$ –ВР является винеровским процессом или коричневым шумом, для которого характерно отсутствие памяти, то естьследующее значение не зависит от всех предыдущих (ряд характеризуется случайной динамикой, рынок эффективен0;
- $H < 0.5$ – антиперсистентный процесс (розовый шум), частая смена направления, при этом система меняется быстрее, чем случайная, наблюдаются частые, но небольшие изменения (временной ряд – фрактал с отрицательной памятью, наблюдается отрицательная корреляция приращений).

Наиболее распространенным методом вычисления показателя Херста является метод нормированного размаха (R/S анализ). Метод анализирует размах суммы отклонений ВР от среднегоарифметического, нормированный путем деления на стандартное отклонение

Для оценки данного параметра была разработана реализация алгоритма R/S. Для данного ряда показатель Херста равен 0.76, что означает, что процесс является персистнтным, а ряд обладает долговременной памятью. Т.е. имеет смысл его моделировать и предсказывать будущие значения.

Вывод: загрузка значений временного ряда и первичная оценка его статистических характеристик, распределения, визуальная оценка графика, гистограммы распределения значений, декомпозиция составляющих и оценка фрактальных свойств является первым шагом для дальнейшего анализа – построения модели и прогнозирования будущих значений. Язык программирования python и библиотека pandas представляют инструмент, позволяющий проводить данный предварительный анализ. Кроме того, целесообразно оценивать фрактальные свойства ряда для выбора решения вопроса о целесообразности дальнейшей работы с рядом и выбора наиболее подходящей модели.

Библиографический список

1. Brockwell, P. a Davis, R. Time Series: Theory and Methods. Praha: Springer-Verlag New York, 2009.
2. Андерсон, Т. Статистический анализ временных рядов / Т. Андерсон. – М.: Мир, 1976. – 751.
3. Weron R. Estimating long-range dependence: finite sample properties and confidence intervals. Hugo Steinhaus Center for Stochastic Methods, Wroc law University of Technology, 50-370 Wroc law, Poland, 2002.

УДК 004.032

ЭНТРОПИЯ КАК МЕРА ОРГАНИЗОВАННОСТИ СИСТЕМ РАСЧЕТОВ ЗА ЖИЛИЩНО-КОММУНАЛЬНЫЕ УСЛУГИ

С.В. Аникеев
Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, asv765@mail.ru

Аннотация. В работе рассматриваются вопросы оценки энтропии систем расчетов за жилищно-коммунальные услуги с учетом наличия неопределенности в значениях качественных и количественных характеристик абонентов.
Ключевые слова: информационные системы, расчет за жилищно-коммунальные услуги, энтропия, Марковские цепи.

ENTROPY AS A MEASURE OF ORGANIZATION OF SYSTEMS OF PAYMENTS FOR HOUSING AND COMMUNAL SERVICES

S.V. Anikeev
Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, asv765@mail.ru

Annotation. The paper considers issues of estimating the entropy of settlement systems for housing and communal services, taking into account the presence of uncertainty in the values of the qualitative and quantitative characteristics of subscribers.
Keywords: information systems, payment for housing and communal services, entropy, Markov chains.

Современное жилищно-коммунальное хозяйство (далее ЖКХ) – одна из крупнейших отраслей экономики и социальной сферы государства. Состояние ЖКХ оказывает прямое воздействие на уровень его социально-экономического развития и социальное самочувствия населения страны.

Особую роль в процессе функционирования системы играют информационно-технологические отношения между институциональными субъектами сферы ЖКХ - совокупность методов поиска, сбора, хранения, обработки, предоставления и распространения

информации с использованием средств вычислительной техники в процессе оказания жилищно-коммунальных услуг (ЖКУ) и расчетов за них.

Применение информационных технологий в сфере ЖКХ является широко используемой и давно устоявшейся практикой.

Среди общей массы информационных систем, применяемых в сфере ЖКХ, особого внимания заслуживают системы расчета и учета платежей за ЖКУ. Такие системы иногда называют "биллинговые".

Ключевой операцией в системах расчета за ЖКУ является определение размера платы и объема потребляемого ресурса или услуги в заданном периоде времени. Основные принципы, определяющие расчет, изложены в ряде нормативно-правовых документов [1]. В соответствии с документами, расчетный период устанавливается равным одному календарному месяцу. В результате анализа рекомендованных формул, расчет объема и стоимости ЖКУ предлагается выполнять в рамках следующей математической модели [2,3].

Для абонентов, не имеющих индивидуальных и групповых приборов учета, расчет месячного объема и стоимости ЖКУ выполнятся по формуле:

$$S_{i,r} = fs(\overline{QN_i}, \overline{N_r}) \quad (1)$$

Здесь:

$\overline{QN_i}$ - вектор (кортеж) количественных характеристик i-го абонента;

$\overline{N_r}$ - вектор (кортеж) количественных характеристик r-го режима потребления.

Переход от значений объема к значениям стоимости представляется, как правило, тривиальным и заключается в умножении полученных значений на нормирующий коэффициент – стоимость единицы объема ЖКУ.

Следует отметить, что расчет для i-го абонента ведется не по всей совокупности возможных режимов потребления R, а по некоторому их подмножеству: $R^{<i>} \in R$. Назовем такое подмножество «активные режимы потребления». Определение элементов подмножества также является вычислительной задачей:

$$R^{<i>} = fr(\overline{QL_i}), \quad (2)$$

где $\overline{QL_i}$ - вектор (кортеж) качественных характеристик i-го абонента.

Не вдаваясь в особенности расчета, отметим, что общая совокупность значений объема и стоимости определяется значениями количественных $\overline{QN}$ и качественных $\overline{QL}$ характеристик абонентов, а также рядом количественных показателей $\overline{N}$, характеризующих режимы потребления ЖКУ:

$$\|S\| = F(\overline{QN}, \overline{QL}, \overline{N}) \quad (3)$$

Зависимость объема и стоимости ЖКУ от значений количественных и качественных характеристик абонентов, а также количественных показателей режимов потребления предъявляет особые требования к точности и актуальности этих значений.

На практике, как правило, проблем с актуальностью и точностью $\overline{N}$ не возникает – соответствующие значения устанавливаются нормативными актами и обычно известны до начала их действия. Совершенно другая ситуация складывается со значениями $\overline{QN}$ и $\overline{QL}$. Можно сказать, что информация о характеристиках абонентов – дефицитный ресурс. Очень часто данные о характеристиках поступают в информационную систему с задержкой и могут

содержать ошибки – отличаться от истинных значений параметров, действующих в предметной области. Так, например, одной из наиболее существенных характеристик абонента является «количество человек, потребителей услуги» – величина, определяющая объем и стоимость холодного и горячего водоснабжения, газоснабжения, вывоза твердых коммунальных отходов и др. Причем данное значение используется даже в случаях, когда расчет ведется по показаниям приборов учета. Источником данных о значении этой характеристики часто является сам абонент, который сообщает сведения об изменения характеристики нерегулярно, а зачастую – не сообщает вовсе.

Для устранения негативных экономических последствий такой ситуации в современных информационных системах предусмотрена операция перерасчета – изменения размера платы за прошлые периоды в случае актуализации характеристики абонента «задним числом». Наличие такой операции позволяет до известной степени снизить остроту проблемы, однако не способствует выявлению и устранению её причин. Кроме того, механизм перерасчетов действует только до определенного порога, так как в случае массового количества рассогласований и, следовательно, недостоверного расчета, в регионе значительно возрастает социальная напряженность по отношению к организации – владельцу информационной системы, что чревато массой негативных последствий.

Таким образом, проблема анализа качества построения информационной системы расчетов за ЖКУ с учетом достоверности имеющейся в ней информации представляется актуальной.

Рассмотрим возможные случаи отображения значения некоторой количественной характеристики абонента в предметной области на соответствующий показатель в базе данных информационной системы. Для упрощения предположим, что показатель может принимать только целые значения (например, «количество человек, потребителей услуги»). Тогда для случаев, когда значение характеристики достоверно неизвестно, об истинном значении мы можем судить по вектору $\overline{p}$, каждый элемент которого задает вероятность того, что характеристика принимает соответствующее значение. Численные значения $\overline{p}$ можно определить статистическими методами, как отношение количества абонентов с заданным значением характеристики к общему количеству абонентов. Так, например, характеристика «количество человек, потребителей услуги» для абонентов г.Рязани имеет следующий вид $\overline{p} = \{0.078, 0.315, 0.259, 0.188, 0.104, 0.037, 0.012, 0.007\}$. Здесь $p_0 = 0.078$ - вероятность того, что значение данной характеристики равно «0», $p_1 = 0.315$ - равно «1» и т.д.

В случае же полной определенности, когда значение характеристики в информационной системе равно k, что также соответствует значению в предметной области:

$$p_j = \begin{cases} 1, j = k \\ 0, j <> k \end{cases}.$$

Таким образом, используя классическую формулу информационной энтропии Шеннона, мы можем рассчитать меру неопределенности для заданной характеристики q:

$$H^{<q>} = -\sum_{j=1}^{n} p_j^{<q>} \log p_j^{<q>} \quad (4),$$

где n - количество всех возможных значений характеристики q.

Рассмотрим простейший случай, в котором значение характеристики q абонента находятся в одном из двух состояний:

$S_0^{<q>}$ - значение характеристики q в предметной области соответствует значению в базе данных информационной системы;

$S_1^{<q>}$ - значение в предметной области отличается от значения в базе данных информационной системы.

$S_0^{<q>}$ и $S_1^{<q>}$ образуют полную группу событий и $P(S_0^{<q>}) + P(S_1^{<q>}) = 1$.

Заметим, что для состояния $S_0^{<q>}$ (в случае полной определенности) $H^{<q>} = 0$, так как $1 \cdot \log 1 = 0$ и $\lim\limits_{p_i^{<q>} \to 0} p_i^{<q>} \cdot \log p_i^{<q>} = 0$. Тогда, учитывая аддитивный характер величины $H^{<q>}$, имеем:

$$H_{\sum}^{<q>} = N^{<S_1>} \cdot H^{<q>}, \tag{5}$$

где $N^{<S_1>}$ – количество абонентов, находящихся в состоянии $S_1^{<q>}$ (не находящихся в состоянии $S_0^{<q>}$).

Учитывая, что статистически вероятность проявляет себя как количество исходов, соответствующих заданному событию, по отношению к общему числу опытов, можем заметить, что $P(S_1^{<q>}) = \dfrac{N^{<S_1>}}{N}$, где N - общее количество абонентов и

$$H_{\sum}^{<q>} = P(S_1^{<q>}) \cdot N \cdot H^{<q>}.$$

Для оценки $P(S_1^{<q>})$ представим рассматриваемый простейший случай как непрерывную Марковскую цепь с двумя состояниями - $S_0^{<q>}$ и $S_1^{<q>}$.

Из теории известно, что для данной системы в стационарном режиме

$$P(S_1^{<q>}) = \frac{\lambda^{<q>}}{\mu^{<q>} + \lambda^{<q>}}, \tag{6}$$

где $\mu^{<q>}$ - интенсивность потока перехода системы из состояния $S_1^{<q>}$ в $S_0^{<q>}$ (значение изменившегося параметра q обрабатывается и сохраняется в базе данных информационной системы);

$\lambda^{<q>}$ - интенсивность потока перехода системы из состояния $S_0^{<q>}$ в $S_1^{<q>}$ (параметр q изменился в предметной области, но данные еще не поступили в информационную систему).

Итоговое соотношение для оценки энтропии системы в целом по параметру q принимает вид:

$$H_{\sum}^{<q>} = \frac{\lambda^{<q>}}{\mu^{<q>} + \lambda^{<q>}} \cdot N \cdot H^{<q>}. \tag{7}$$

Вышесказанное в полной мере справедливо также для любой качественной характеристики, принимающей для каждого абонента в заданный момент времени только одно из предопределенных значений, а также для количественных характеристик, принимающих дробные значения в относительно широком диапазоне (например, размер площади жилого

помещения). Для последних, однако, с целью практического применения целесообразно перейти к диапазонам значений с оценкой вероятности нахождения характеристики в заданном диапазоне.

Тогда значение энтропии системы в целом может быть вычислено как:

$$H_{\sum} = \sum_{q \in Q} H_{\sum}^{<q>}, \quad (8)$$

где Q – множество количественных и качественных характеристик, используемых для расчета.

Выводы

Приведенные соотношения позволяет оценить меру неопределенности системы расчетов за ЖКУ в целом с учетом наличия в базе данных информационной системы устаревших (неактуальных) сведений.

Использование цепей Маркова позволяет анализировать широкий класс информационных систем, используя в качестве исходных данных плотности потоков изменения параметров в предметной области, обработки и сохранения информации в БД информационной системы. Соответствующие значения могут быть получены расчетным путем в результате анализа уже накопленных данных.

Предложенный подход позволяет выполнять сравнительный анализ архитектур построения систем расчетов за ЖКУ, руководствуясь принципом, что системы с меньшим значением энтропии являются более организованными по сравнению с системами, энтропия которых при прочих равных принимает большие значения.

Библиографический список

1. Постановление Правительства РФ от 06.05.2011 № 354 (ред. от 15.09.2018) "О предоставлении коммунальных услуг собственникам и пользователям помещений в многоквартирных домах и жилых домов" (вместе с "Правилами предоставления коммунальных услуг собственникам и пользователям помещений в многоквартирных домах и жилых домов").

2. Аникеев, С.В. О математической модели ведения расчета за жилищно-коммунальные услуги / С.В. Аникеев, М.Г. Костиков // Математическое и программное обеспечение вычислительных систем: Межвуз.сб.науч.тр. / Под ред. А.Н. Пылькина – Рязань (РГРТУ), июнь 2013. – 152 с. С.72-81

3. Аникеев, С.В. Математическая модель определения активных услуг на основе матричной алгебры / С.В. Аникеев, М.Г. Костиков, А.В.Маркин // Вестник Рязанского государственного радиотехнического университета № 4 (выпуск 54) 2015 г. Часть 1, РГРТУ. 2015. ISSN: 1995-4565. 164 с. С.139-143.

УДК 004.4'2; ГРНТИ 20.23

ОСОБЕННОСТИ РАЗРАБОТКИ ПРИЛОЖЕНИЯ «ТАБЛО ДЛЯ ГОЛОСОВАНИЯ» С ИСПОЛЬЗОВАНИЕМ ГРАФИЧЕСКИХ КОМПОНЕНТ

Е.С. Щенёв

Рязанский государственный радиотехнический университет, Российская Федерация, Рязань,Jheka1235317@gmail.com

Аннотация. В работе рассматриваются особенности разработки приложения с помощью графических модулей. Проводится анализ разработки приложения, структуры программы, представлена упрощенная схема работы приложения, дана краткая аннотация программы с подробным описанием возможностей, достоинств, особенностей разработки.

Ключевые слова: приложение, графический модуль, процедура, переменные, графический интерфейс.

FEATURES OF THE DEVELOPMENT OF THE APPLICATION "SCOREBOARD FOR VOTING" USING GRAPHICAL COMPONENTS

E.S. Schenev
Ryazan State Radio Engineering University,
Russia, Ryazan,Jheka1235317@gmail.com

The summary. This article discusses the features of application development using graphical modules. The analysis of application development, program structure, represents a simplified scheme of the application was given a brief annotation of the program with a detailed description of features, advantages, and features of the development.
Keywords: application, graphical module, procedure, variables, graphical interface.

В современном мире во всех сферах человеческой деятельности активно применяются информационные системы, которые упрощают труд людей. Стремительные темпы глобализации за последние десять лет были в основном вызваны разработками в области информационных и коммуникационных технологий. Количество пользователей телекоммуникационных устройств и всевозможных вычислительных систем становится все больше и больше. Спрос на ИТ-специалистов растет в целом ряде отраслей. Труд программиста приобретает новые обороты и требует новых навыков и знаний. Создаются новые языки программирования, которые вытесняют старые.

Профессионалы в области программирования всегда тесно сотрудничают с заказчиками для модернизации существующих или создания новых систем. Они могут адаптировать типовое программное обеспечение и интегрировать его в существующие системы.

Для разработки приложений с использованием графических компонент используются языки программирования, которые содержат широкий набор типов, констант и функций для управления графическим режимом работы экрана, объединенных в стандартный библиотечный модуль GRAPH. С помощью подпрограмм, входящих в данный модуль, создаются сложные графические изображения. Для вывода текстовых надписей на графический экран могут быть использованы штриховые и матричные шрифты.

Подпрограммы модуля GRAPH могут поддерживать различные типы аппаратных графических средств. Настройка на конкретные технические средства осуществляется с помощью специальных программ – драйверов, поставляемых вместе с данным модулем.

Приложение «Табло для голосования» разрабатывалось с использованием данного библиотечного модуля. Программа имеет простой и понятный интерфейс, которым будет удобно пользоваться человеку независимо от его уровня владения компьютером. Разработанное приложение позволяет вводить внешние данные удобным образом через интерфейс пользователя и системы меню.

Анализ разработки приложения

Для реализации приложения «Табло для голосования» необходимо разработать информационную систему, которая будет содержать информацию о пользователях системы, вопросах голосования, с отведенным временем для голосования и графический интерфейс приложения, с помощью которого осуществляется ввод информации.

При проектировании информационной системы используются записи для хранения данных о пользователях и вопросах голосования. Сами записи будут храниться в типизированных файлах на диске компьютера. Все операции над ними будут производиться при помощи процедур, содержащихся в модулях.

Так как в программе обрабатываются личные данные пользователей, то необходимо реализовать авторизацию пользователей. Доступ к данным пользователей и их редактирова-

ние, а также к вопросам для голосования было решено предоставить администратору системы. Пользователь может создавать и изменять вопросы, а также редактировать информацию в собственном профиле.

Графический интерфейс будет спроектирован при помощи графического модуля Graph и разбит на несколько частей: окно входа, окно регистрации, окно администратора, окно пользователя, табло для голосования. Переход между ними будет осуществляться при помощи специальных процедур инициализации окон.

Процедуры и переменные, применяемые только в одном окне приложения, будут располагаться в отдельных модулях, чтобы предотвратить ошибки перехода между окнами.

Действия мыши и клавиатуры будут обрабатываться с помощью встроенных процедур модуля Graph, таких как onKeyDown и т.д.

Разработка приложения

Разработка приложения начинается с разработки упрощенной схемы работы приложения. После запуска приложения открывается окно авторизации (рис. 1), из которого можно попасть в окно регистрации (рис. 2), окно администратора и окно пользователя.

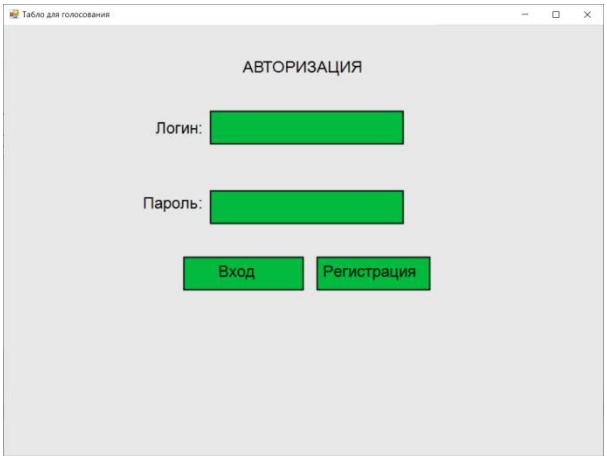


Рис. 2. Окно авторизации

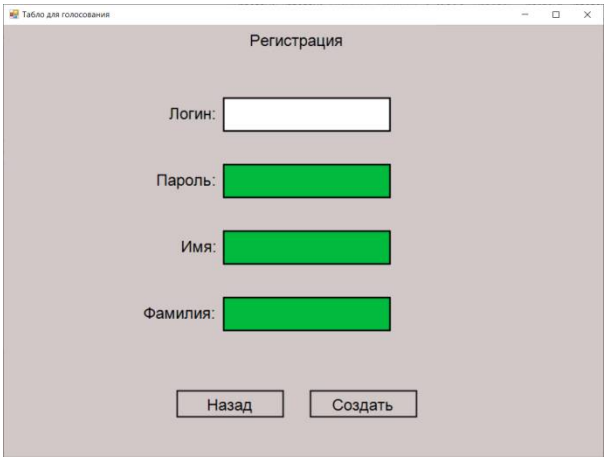


Рис. 2 Окно регистрации

Окна администратора и пользователя являются основными рабочими окнами приложения (рис. 3-4).

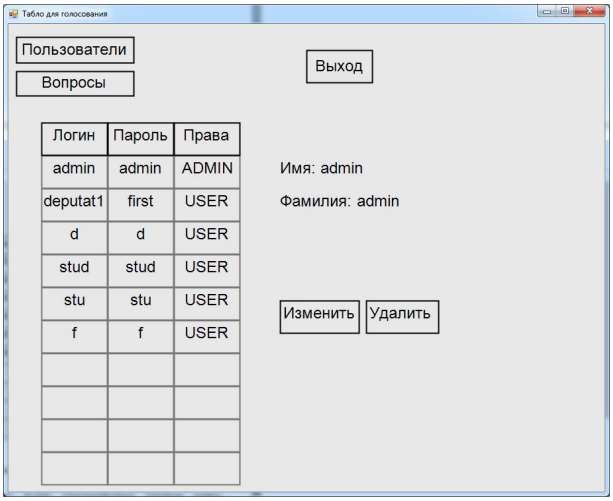


Рис. 3. Окно администратора

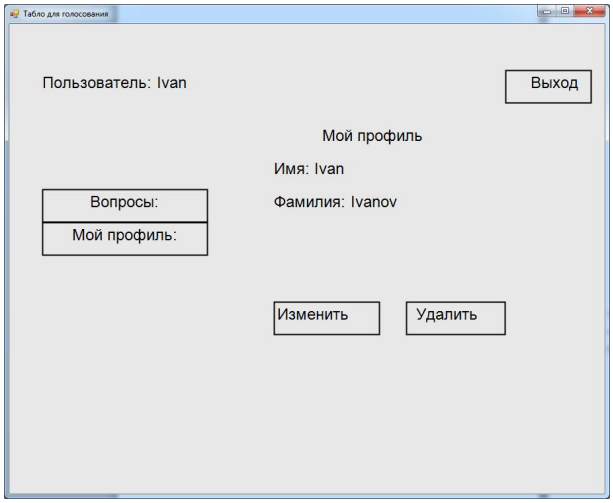


Рис. 4. Окно пользователя

Разработка программы была разбита на несколько этапов:

– создание двух модулей для локализации всех процедур и функций, связанных с записями пользователей и вопросов для голосования;

– создание двух типов записей: содержащего информацию о пользователе и содержащего информацию о вопросах;

– создание процедур, которые производят запись в файлы записей пользователей и вопросов для голосования, а также процедур для их удаления;

– создание процедур для получения массивов всех записей пользователей и вопросов для голосования для отображения полных списков пользователей и вопросов в графическом интерфейсе в окне администратора и в разделе «Вопросы» в окне пользователя;

– создание системы обработки информации о пользователях, которые ответственны за проведение процедуры голосования и вопросах, выносимых на голосование;

– разработка вспомогательных процедур и функций, помогающих привести эти данные в удобный вид для отображения в графическом интерфейсе;

– разработка графического интерфейса. Реализация перехода между окнами осуществляется с помощью методов инициализации, которые переопределяют процедуру обработки событий мыши и клавиатуры, определяет начальные значения переменных данного модуля и запускают процедуру рисования самого окна.

Краткая аннотация программы с описанием возможностей, достоинств, особенностей разработки

Написанная программа позволяет выдавать на экран дисплея табло для голосования, аналогичное табло в Верховном Совете.

Программа имеет простой и понятный интерфейс, которым удобно пользоваться любому человеку, независимо от его уровня владения компьютером. Данная программа позволяет вводить данные удобным образом через интерфейс пользователя и систему меню. В программе предусмотрены два режима ввода данных о голосовании: вручную с помощью функциональных клавиш и автоматически с помощью датчика случайных чисел, из файла с заранее сгенерированными данными.

При запуске программы, пользователю необходимо войти в личный кабинет, либо зарегистрироваться, если созданной страницы нет. В профиле и структуре программы будут учитываться возможности пользователя в соответствии с выданными администратором правами. Если у пользователя нет учетной записи в системе, то ее можно создать, нажав на кнопку регистрации в окне авторизации пользователя (рис. 1) и, перейдя в специальное окно регистрации пользователя создать новую учетную запись (рис. 2).

После входа будет предложено просмотреть существующие вопросы, проголосовать, если участие необходимо и не осуществлено до сих пор. Права администратора добавляют возможности удаления вопросов, редактирования их у каждого пользователя, а также изменение выданных привилегий (рис. 5). Обычный пользователь может создавать и изменять только собственные вопросы, а также редактировать информацию в собственном профиле (рис. 6).

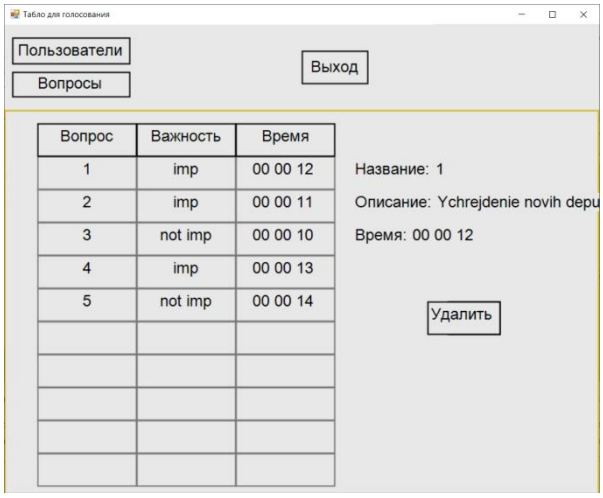


Рис. 5. Окно администратора «Просмотр вопросов для голосования»

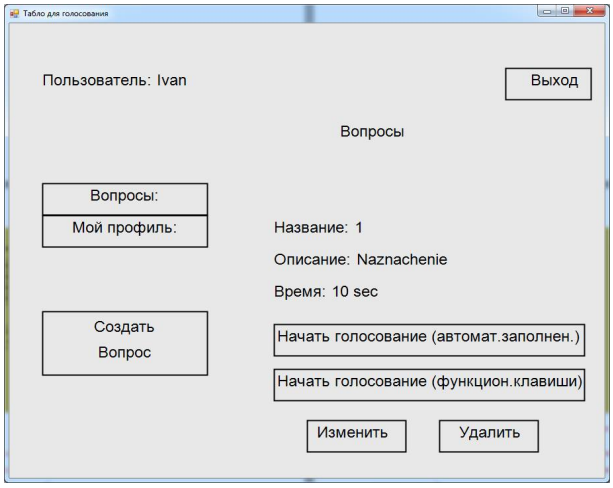


Рис. 6. Выбор конкретного вопроса голосования пользователем

В программе реализован вход в предложенные разделы профиля каждого пользователя, который принимает участие в голосовании. Для того чтобы получить результаты голосования по выбранному вопросу, необходимо нажать одну из предложенных кнопок, а именно: начать голосование с помощью автоматического заполнения (в режиме тестирования и отладки системы) и начать голосование с заполнением с помощью вспомогательных клавиш, назначение которых будет указано на открывшемся окне.

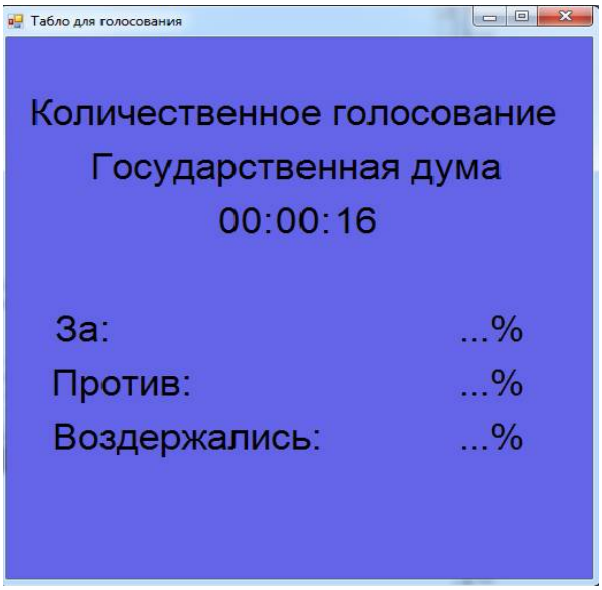


Рис. 7. Окно голосования по вопросу

Рис. 8. Окно результатов голосования

Отличительной особенностью запуска голосования является выбор конкретного режима работы, а именно, с помощью функциональных клавиш (F1-За, F2–Против, F3–Воздержался) или с помощью автоматического заполнения, из заранее созданного файла, с генерированными данными. Нажав на кнопку Начать голосование, открывается окно голосования, в котором отображается информация о голосовании по текущему вопросу. В окне присутствует таймер времени по каждому вопросу (рис. 7)

По окончании времени голосования выводится окно с результатами голосования и заключением по данному вопросу о принятии или непринятии решения (рис. 8)

По результатам голосования будет принято решение по самому вопросу. Если число проголосовавших «за» будет большим, чем сумма голосов, проголосовавших «против» и «воздержались», то решение будет принято. В обратном же случае решение будет не принято.

Таким образом, было разработано графическое приложение «Табло для голосования», которое имеет простой и понятный интерфейс. Графический интерфейс позволяет в режиме онлайн отображать оставшееся время, отведенное на голосование, по окончании которого выводятся результаты голосования.

Библиографический список

1. Голицина О. Л., Попов И. И. Основы алгоритмизации и программирования: Учебное пособие. – М.: ФОРУМ, 2015.–432с.
2. Зыков С.В. Программирование. Объектно-ориентированный подход. Учебник и практикум. – М.: Юрайт, 2017.–155 с.
3. Керниган Б., Пайк Р. Практика программирования. – М.: Вильямс, 2017.–288 с.

СЕКЦИЯ «ЭВМ И СИСТЕМЫ»

УДК 681.3.06; ГРНТИ 28.17.23

МОДЕЛИРОВАНИЕ ИНФОРМАЦИОННЫХ СТРУКТУР ДЛЯ ИССЛЕДОВАНИЯ АЛГОРИТМОВ ВЕРИФИКАЦИИ БАЗ ДАННЫХ

А.И. Баранчиков, Н.З. Нгуен
Рязанский государственный радиотехнический университет имени В.Ф. Уткина,
Российская Федерация, Рязань, nguyenngocduong1991@gmail.com

Аннотация. В этой работе разработан алгоритм для создания тестовой базы данных, позволяющей моделировать информационную структуру, на которых могут проверяться алгоритмы верификации базы данных. А также приведен пример выполнения данного алгоритма для заданной схемы базы данных, чтобы создать тестовую базу данных.
Ключевые слова: реляционная база данных, тестовая база данных, алгоритм, моделирование, верификация, функциональная зависимость, информационная структура, предметная область.

MODELING INFORMATION STRUCTURES FOR RESEARCH OF ALGORITHMS FOR DATABASE VERIFICATION

A.I. Baranchikov, N.D. Nguyen
Ryazan State Radio Engineering University named after V.F. Utkin,
Russia, Ryazan, nguyenngocduong1991@gmail.com

The summary. In this paper, we developed an algorithm for creating a test database that allows us to model the information structure on which the algorithms for database verification can be tested. And also an example of the work of this algorithm for the existing database schema to create a test database is given.
Keywords: relational database, test database, algorithm, modeling, verification, functional dependence, information structure, subject area.

Введение

Как известно, моделирование информационных структур не только способствует повышению производительности работы базы данных (БД) за счет более быстрого извлечения данных с обеспечением целостности и гибкости в процессе эксплуатации, но и позволяет проверить содержащиеся в ней данные на соответствие реальности, т.е. их достоверности, на которой основаны алгоритмы верификации базы данных [1].

Целью данной работы является разработка алгоритма для создания тестовой БД, которая позволяет моделировать информационную структуру БД. С помощью этого алгоритма можно получить все семантические зависимости данных и дополнительные ограничения на предметную область.

Постановка задачи

При создании тестовой БД необходимо учитывать ряд специфических критериев, которые определяют класс предметной области, зависящий от сложности ее семантической организации и количества присущих ей атрибутов, учитывающих распределенность данных и их уровень. Таким образом, тестовая БД должна обладать следующими свойствами:

1. Все схемы тестовой БД должны относиться к одной предметной области (ПО) и они должны обладать одним общим множеством атрибутов $\boldsymbol{U}$.

2. Все схемы тестовой БД должны находиться в третьей нормальной форме (3НФ) и имеют все ограничения ПО, не связанные с целостностью данных

3. В тестовой БД должны существовать пары схемы отношений R и S, где

$$R_i = \{X_1, X_2, \ldots, X_p\}, \qquad R_j = \{X_1, X_2, \ldots, X_q\}.$$

И должно соблюдаться условие

$$\boldsymbol{U} = R_i \cup R_j = X_1 \cup X_2 \cup \ldots \cup X_p \cup X_1 \cup X_2 \cup \ldots \cup X_q.$$

Данное условие говорит о том, что эти схемы баз данных должны включать те и только те атрибуты, которые изначально присутствуют в множестве атрибутов $\boldsymbol{U}$.

4. Множество функциональной зависимости (ФЗ) F над U должно быть минимальным и редуцированным.

5. Должно задаваться неизбыточное множество ФЗ F, представленных в виде кольцевого множества составных функциональных зависимостей (СФЗ) вида $(X_1, X_2, \ldots X_k) \rightarrow Y$, где каждая СФЗ представляет отдельный класс эквивалентности $E_F(X)$, а множество СФЗ - множество классов эквивалентности $\bar{E}_F$, а количество классов эквивалентности $|\bar{E}_F|$ соответствует количеству схем $|\boldsymbol{R}|$ корректно спроектированной схемы базы данных [2, 3].

Построение алгоритма для создания тестовой БД

Пусть задана схема реляционной базы данных (РБД) $\boldsymbol{V} = \{V_1, V_2, \ldots, V_k\}$ где k – число отношений в схеме $\boldsymbol{V}$; и множество ФЗ F. Для создания тестовой БД, которые обладают вышеуказанные свойства, необходимо реализовать следующие этапы:

Этап 1. Определение и анализ ПО для создания модели предметной области.

Этап 2. Определение общего множества атрибутов: $U = V_1 \cup V_2 \cup \ldots \cup V_k$.

Этап 3. Нахождение неизбыточного и редуцированного покрытия для заданного множества ФЗ F.

Этап 4. Нахождение классов эквивалентности $\bar{E}_F$ для множества F.

Этап 5. Нахождение минимального множества ФЗ F.

Этап 6. Создание кольцевого покрытия в виде множества СФЗ на основе найденного множества классов эквивалентности $\bar{E}_F$.

Этап 7. Нахождение минимального редуцированного кольцевого покрытия в виде множества СФЗ.

Этап 8. Построение схемы РБД $R = \{R_1, R_2, \ldots, R_n\}$ на основе минимального редуцированного множества СФЗ.

Этап 9. Обработка полученной схемы РБД для удовлетворения 3НФ.

Тестовая БД создается на основе классического алгоритма синтеза схем РБД, который заключается в нахождении неизбыточного и редуцированного покрытия для заданного множества ФЗ F и последующем создании минимального редуцированного кольцевого покрытия в виде СФЗ с помощью классов эквивалентности $\bar{E}_F$. В конце происходит синтез схем РБД $\boldsymbol{R} = \{R_1, R_2, \ldots, R_n\}$ на основе множества СФЗ. Разработанный алгоритм завершается после обработки полученных схем баз данных для удовлетворения 3НФ. Полученная в результате схема баз данных $\boldsymbol{R} = \{R_1, R_2, \ldots, R_n\}$ является тестовой БД.

На первом этапе определяется и анализируется ПО для создания модели предметной области. Предметная область – это часть реального мира, представленная совокупностью реальных и абстрактных объектов, которые характеризуется свойствами. Для создания РБД нужно сначала выполнить анализ ПО и создать ее модель. Основой для анализа предметной области служат документы, отражающие ПО, и информация, которую можно получить от экспертов этой предметной области. Анализ ПО в базах данных позволяет выявить объекты и атрибуты сущностей, определить начальные требования для функциональности и границы проекта. С помощью модели ПО можно установить в БД семантические зависимости и связи между объектами предметной области [4].

На втором этапе из заданной схемы реляционной базы данных $V=\{V_1, V_2, \dots, V_k\}$ определим общее множество атрибутов: $U= V_1 \cup V_2 \cup \dots \cup V_k$;

На третьем этапе выполняется поиск неизбыточного покрытия для заданного множества F, затем происходит этап редуцирования полученного покрытия. Пусть F и G – два множества ФЗ, множество G является покрытием для множества F, если $G \equiv F$. Множество G называется неизбыточным, если в G не существует ФЗ $X \to Y$, такой, что $F - \{X \to Y\} \vDash X \to Y$.

Для того, чтобы найти неизбыточное покрытие G для множества ФЗ F, необходимо найти множество G, такое, что $G \equiv F$ и G неизбыточно.

Множество ФЗ F называется редуцированным, если каждая ФЗ из F редуцирована, то есть, в каждой ФЗ не содержатся посторонние атрибуты, как в левой части, так и в правой части. Следует отметить, для нахождения редуцированного покрытия нужно вначале удалить все посторонние атрибуты из левой части ФЗ, а затем из правой.

На четвертом этапе выполняется поиск классов эквивалентности.

Пусть X – множество атрибутов, $X \subseteq R$. Класс эквивалентности $E_F(X)$ называется множеством ФЗ из F с левой частью, эквивалентной X. Обозначим $\bar{E}_F$ множество: $\{E_F(X) | X \subseteq R$ и $E_F(X) \neq \emptyset\}$, а $|\bar{E}_F|$ – количество классов эквивалентности в R. Если в F не существует ФЗ с левой частью, эквивалентной X, то $E_F(X)$ пусто. Пусть $F= \{A \to BC, B \to A, AD \to E\}$. Тогда $\bar{E}_F$ имеет вид: $E_F(A)= \{A \to BC, B \to A\}$, $E_F(AD)= \{AD \to E\}$.

На пятом этапе выполняется минимизация множества ФЗ F. Множество ФЗ F является минимальным множеством, если F не содержит избыточных ФЗ, то есть оно содержит не больше ФЗ, чем любое его эквивалентное множество ФЗ.

Пусть G – неизбыточное неминимальное множество ФЗ. Для вычисления минимальных покрытий для G нужно найти все пары ФЗ $X \to Z, Y \to Q$, для которых $X \leftrightarrow Y$ и $X \dot{\to} Y$. Если такие зависимости найдены в G, то их необходимо заменить одной ФЗ $Y \to ZQ$,

где:

X, Y, Z, Q – подмножества входного множества атрибутов;

$X \leftrightarrow Y$ –два множества атрибутов X и Y эквивалентны относительно множества F, если $F \vDash X \to Y$ и $F \vDash Y \to X$;

$X \dot{\to} Y$ – прямая определяемость (X прямо определяет Y), это означает, что можно найти неизбыточное покрытие G для множества F, причем в G содержит ФЗ $X \to Y$, которая выводит только из множества ФЗ, принадлежащих $F - E_F(X)$. То есть $(F - E_F(X)) \vDash X \to Y$.

Например, пусть $F =\{A \to CD, AB \to E, B \to I, DI \to J, J \to AB\}$. Имеем пары ФЗ $AB \to E$, $J \to AB$, для которых $AB \leftrightarrow J$ и $AB \dot{\to} J$ относительно F. Можно заменить $AB \to E, J \to AB$ на одной ФЗ $J \to ABE$. Тогда получаем эквивалентное минимальное множество $F'=\{A \to CD, B \to I, DI \to J, J \to ABE\}$.

На шестом этапе строится кольцевое покрытие в виде множества СФЗ для F. СФЗ имеет вид: $(X_1, X_2, \dots X_k) \to Y$, где множество $(X_1, X_2, \dots X_k)$ называется левой частью, Y – правой частью. Для заданного множества F- зависимостей G можно найти кольцевое покрытие с не более чем $|\bar{E}_F|$ СФЗ, где F– неизбыточное покрытие множества G. Все F- зависимости из одного $E_F(X)$ можно объединить в одну СФЗ. Каждая левая часть из $e_F(X)$ есть левое множество СФЗ, а правая является объединением всех правых частей ФЗ из $E_F(X)$ [5].

Например, пусть $F=\{A \to BC, B \to AD, AE \to I, BE \to IJ\}$. Тогда кольцевым покрытием множества F служит $G= \{(A, B) \to ABCD, (AE, BE) \to IJ\}$.

На седьмом этапе выполняется минимизация и редукция полученных СФЗ для входного множества ФЗ F. Поскольку минимальность СФЗ выполняется за счет минимизации самого множества F, выполненного на пятом этапе, то на данном этапе полученные СФЗ необходимо лишь редуцировать. Для редуцирования СФЗ необходимо выполнять следующие шаги:

– удалить все перемещаемые атрибуты в левой части СФЗ. Под *перемещаемыми* атрибутами понимаются атрибуты, принадлежащие левым множествам СФЗ, при перемещении которых в правое множество той же СФЗ не нарушается эквивалентность.

– удалить все посторонние атрибуты в правой части СФЗ. Под *посторонними* атрибутами понимаются атрибуты принадлежащие множеству ФЗ или СФЗ, при удалении которых не изменяется замыкание данного множества зависимостей [6].

На восьмом этапе строится схема баз данных $\boldsymbol{R} = \{R_1, R_2, \ldots, R_n\}$ на основе минимального редуцированного множества СФЗ. Каждая СФЗ вида $(A_1, A_2, \ldots, A_m) \rightarrow B$ строится схема отношений $R_i = A_1\, A_2 \ldots A_m\, B$ с выделенным ключом $K_i = A_1\, A_2 \ldots A_m$.

На последнем этапе для удовлетворения 3НФ необходима дополнительная обработка полученного набора отношений.

Алгоритм ***Create_Test_DB*****,** приведен ниже, позволяет создать тестовую БД.

Create_Test_DB

Вход:

- схема реляционной базы данных $\boldsymbol{V} = \{V_1, V_2, \ldots, V_k\}$ где V_i – имя отношений, принадлежащих схемам V.

$V_i = \{A_{i1}, A_{i2}, \ldots, A_{in}\}$, где Aij – имя атрибута.

k– количество отношений, принадлежащих базе данных $\boldsymbol{V}$.

n – количество атрибутов в схеме отношения V_i.

- множество функциональной зависимости $F = \{X_1 \rightarrow Y_1, X_2 \rightarrow Y_2 \ldots X_n \rightarrow Y_n\}$

Выход*:* тестовая база данных $\boldsymbol{R}$

***Create_Test_DB* (*V*, *F*)**

Begin

Определить предметную область и анализировать ее для создания модели БД;

Определить общее множество атрибутов:

$\boldsymbol{U} = V_1 \cup V_2 \cup \ldots \cup V_k$;

Поиск неизбыточного покрытия для множества F:

G:=F;

for каждая ФЗ $X \rightarrow Y$ из G **do**

if $(F - \{X \rightarrow Y\} \vDash X \rightarrow Y)$ **then** $F := F - \{X \rightarrow Y\}$

Редуцирование слева полученного неизбыточного покрытия для F:

$G = F$;

for каждая ФЗ $X \rightarrow Y$ из G **do**

for каждый атрибут A из X **do**

if $(F \vDash (X - A) \rightarrow Y)$ **then**

удалить A из X в $X \rightarrow Y$ из F.

Редуцирование справа для полученного множества F:

$G = F$;

for каждая ФЗ $X \rightarrow Y$ из G **do**

for каждый атрибут A из Y **do**

if $(F - \{X \rightarrow Y\} \cup \{X \rightarrow (Y - A)\} \vDash X \rightarrow A$ **then**

удалить A из Y в $X \rightarrow Y$ из F.

Поиск классов эквивалентности E_f для F:

$\bar{E}_F = \emptyset$;

for каждая ФЗ $X \rightarrow Y$ из F **do**

$Left\_part = Left\_part + X$;

for каждое множество $X_i \in Left\_part$ **do**

begin

for каждая ФЗ $U \rightarrow V$ из F **do**

if $U \leftrightarrow X_i$ **then** $E_F(X_i) = E_F(X_i) + U \rightarrow V$;

end;
$\bar{E}_F = \bar{E}_F + E_F(X_i)$;
Минимизация множества ФЗ F:
for каждая $E_F(X)$из $\bar{E}_F$ **do**
for каждая $Y \to U$ из $\bar{E}_F$ **do**
for каждая $Z \to V \neq Y \to U$ из $\bar{E}_F$ **do**
if F $\vDash Y \to Z$ **then** Заменить $Y \to U$ и $Z \to V$ на $Z \to UV$ в F;
Построение кольцевого покрытия в виде множества СФЗ:
for каждый $E_F(X)$из $\bar{E}_F$ **do**
for каждая ФЗ $U_i \to V_i$ $(1 \le i \le m)$ из $E_F(X)$ **do**
$CF_j = (U_1, U_2, ..., U_m) \to V$, где $V = V_1 \cup V_2 \ldots \cup V_m$ и $1 \le j \le n$;
Редуцирование полученного множества СФЗ:
for каждая СФЗ CF_j $(1 \le j \le n)$ вида $(U_1, U_2, ..., U_m) \to V$ **do**
for каждый атрибут A из U_i, $1 \le i \le m$ **do**
if A — перемещаемый атрибут **then**
begin
$U_i = U_i - A$; $V = V \cup A$;
end;
if B — посторонний атрибут из V **then** $V = V - B$;
Построение схемы баз данных $R = \{R_1, R_2, ..., R_n\}$:
for каждая СФЗ CF_j $(1 \le j \le n)$вида $(U_1, U_2, ..., U_m) \to V$ **do**
$R_j = U_1 U_2 ... U_f V$; $\boldsymbol{R} = \boldsymbol{R} \cup R_j$;
Обработка полученной схемы базы данных $\boldsymbol{R} = \{R_1, R_2, ..., R_n\}$;
end.

Пример работы алгоритма

Пусть дана база данных $\boldsymbol{V}=\{V_1, V_2, V_3\}$,
где $V_1 = A\ B_1\ B_2\ C_2\ D$; $V_2 = B_2\ C_1\ D\ E\ I_1\ I_2$; $V_3 = A\ I_1\ I_2\ I_3$;
и множество ФЗ $F = \{A \to B_1\ B_2\ I_1\ I_2\ I_3;\ B_1\ B_2\ C_1 \to A\ C_2\ D\ E\ I_1\ I_2\ I_3;\ B_1\ B_2\ C_2 \to A\ C_1\ D\ E\ I_1\ I_2\ I_3;\ A \to C_1\ C_2\ D\ E;\ E \to I_1\ I_2\ I_3;\ I_1\ I_2 \to I_3;\ I_2\ I_3 \to I_1;\ I_1\ I_3 \to I_2\}$
По алгоритму ***Create_Test_DB*** **(*V*, *F*)**:
Определение общего множества атрибутов:
$\boldsymbol{U} = V_1 \cup V_2 \cup ... \cup V_k = A\ B_1\ B_2\ C_2\ D \cup B_2\ C_1\ D\ E\ I_1\ I_2 \cup A\ I_1\ I_2\ I_3 = A\ B_1\ B_2\ C_1\ C_2\ D\ E\ I_1\ I_2\ I_3$
Поиск неизбыточного редуцированного покрытия G для F:
$G = \{A \to B_1\ B_2\ C_1\ C_2\ D\ E;\ B_1\ B_2\ C_1 \to A;\ B_1\ B_2\ C_2 \to A;\ D \to B_1\ B_2\ E;\ E \to I_1\ I_2\ I_3;\ I_1\ I_2 \to I_3;\ I_2\ I_3 \to I_1;\ I_1\ I_3 \to I_2\}$
Преобразуя к кольцевому покрытию в виде множества СФЗ, затем редуцируя, имеем:
$G = \{(A, B_1\ B_2\ C_1, B_1\ B_2\ C_2) \to D\ E;\ (B_1\ B_2, D) \to E;\ (E) \to I_1\ I_2\ I_3;\ (I_1\ I_2, I_2\ I_3, I_1\ I_3)\}$
Преобразуя каждую СФЗ в схеме отношения с выделенными ключами, получаем $\boldsymbol{R}=\{R_1, R_2, R_3, R_4\}$.
$R_1 = A\ B_1\ B_2\ C_1\ B_1\ B_2\ C_2\ D\ E$ с множеством ключа $K_1 = \{A, B_1\ B_2\ C_1, B_1\ B_2\ C_2\}$;
$R_2 = B_1\ B_2\ D\ E$ с множеством ключа $K_2 = (B_1\ B_2, D)$
$R_3 = E\ I_1\ I_2\ I_3$ с множеством ключа $K_3 = \{E\}$;
$R_4 = I_1\ I_2\ I_3$ с множеством ключа $K_4 = \{I_1\ I_2, I_2\ I_3, I_1\ I_3\}$;
$R_1 \cup R_3 = A\ B_1\ B_2\ C_1\ B_1\ B_2\ C_2\ D\ E \cup E\ I_1\ I_2\ I_3 = \boldsymbol{U}$.

Заключение

Разработан алгоритм для создания тестовой БД, позволяющий моделировать информационную структуру, на которых могут проверяться алгоритмы верификации базы данных.

Тестовая база данных создается с учетом ряда специфических критериев, которые определяют класс предметной области, зависящий от сложности ее семантической организации и количества присущих ей атрибутов. Все схемы отношений тестовой БД находятся в 3НФ с минимальными и редуцированными множествами ФЗ и СФЗ.

Библиографический список

1. Кузнецов С.Д. Основы баз данных. − М.: Бином, 2017. – 488 с.
2. Мейер Д. Теория реляционных баз данных. Пер. с англ. − М.: Мир, 1987. – 608 с.
3. Хомоненко А. Базы данных. –М.: Бином-Пресс, 2007.–736 с
4. Крёнке Д. Теория и практика построения баз данных, 8-е изд. −СПб.: Питер, 2003. − 800 с.
5. Баранчиков А.И. Синтез информационных структур хранения данных на основе анализа предметных областей. − Рязань: ГУП РО Рязанская областная типография, 2014. − 231 с.

УДК 004.42; ГРНТИ 20.53

АНАЛИЗ МЕТОДОВ ПЕРЕХВАТА ФУНКЦИЙ X86-СОВМЕСТИМЫХ WINDOWS ПРИЛОЖЕНИЙ

Д.Н. Егоров
Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, dmitryegorov48@gmail.com

Аннотация. В статье описаны виды перехвата функций. Рассматриваются основные методы перехвата функций x86-совместимых Windows приложений. Приведены примеры использования данных методов.
Ключевые слова: перехват функций, функция-петля, соглашение о вызове.

FUNCTION INTERCEPTION ANALYSIS OF X86-COMPATABLE WINDOWS APPLICATIONS

D.N. Egorov
Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, dmitryegorov48@gmail.com

The summary. Function interception types are given. Fundamental function interception methods of x86-based Windows applications are provided with examples.
Keywords: function interception, detour function, calling convention.

Перехват функций необходим в случаях, когда требуется получить управление программой с целью наблюдения за ходом её выполнения или его изменения. Перехват функций используется для расширения функциональности приложения и исправления ошибок, а также для получения сведений о производительности.

Существует два вида перехвата функций:

- статический;
- динамический.

Статический перехват осуществляется перед загрузкой приложения в оперативную память, то есть до начала выполнения программы. Такой подход подразумевает внесение изменений в содержимое файла, который в последствие будет загружен операционной системой.

Динамический перехват осуществляется уже непосредственно при выполнении приложения, когда содержимое исполняемого файла было загружено в память. Для внесения изменений необходимо получить доступ к виртуальному адресному пространству, в котором приложение работает.

Существует несколько методов перехвата функции. Все они, так или иначе, сводятся к тому, чтобы получить управление программой в момент вызова перехватываемой функции. Для этого имеются следующие способы:

1. перехват в вызывающей функции;
2. перехват в вызываемой функции;
3. перехват через таблицу виртуальных функций;
4. перехват через таблицу импорта.

Перехват в вызывающей функции подразумевает изменение адреса перехватываемой функции в момент её вызова. Так как функция может вызываться из множества различных мест, то для успешного перехвата необходимо провести изменения во всех точках вызова. Данный подход целесообразно использовать только в том случае, когда требуется осуществить перехват в небольшом количестве мест. Как правило, только в одной, двух функциях, в то время как во всех других точках вызова перехвата не будет.

Для того, чтобы выполнить такой перехват, необходимо знать как происходит вызов функции. Рассмотрим вызов следующей функции:

```
void Foo() { /* код функции */ }
```

Функция *Foo* не принимает аргументов и ничего не возвращает. Допустим, она находится по адресу 0x41001001. Вызов такой функции осуществляется простым выполнением инструкции call:

```
  Адрес   | Код операции | Мнемоника инструкции
41000000 | E8 FC0F0000  | call 41001001
41000005 | ...          | ...
```

Для перехвата достаточно изменить адрес назначения инструкции *call* на адрес нашей функции, например 0x80F00F00:

```
  Адрес   | Код операции | Мнемоника инструкции
41000000 | E8 FB0EF03F  | call 80F00F00
41000005 | ...          | ...
```

Стоит обратить внимание на то, что в данном примере используется вызов относительно адреса следующей инструкции (41000005 + FFC = 41001001). Если приложение имеет 32-битную разрядность, то максимальное значение смещения полностью покрывает диапазон адресов доступных для приложения. В случае с 64-битной разрядностью может потребоваться вызов по абсолютному смещению, для чего необходимо в пределах 32-битного смещения найти свободное место для записи вызова по абсолютному адресу:

```
      Адрес      | Код операции  | Мнемоника инструкции
0000000041000000 | E9 FB4F0000   | jmp 41005000
0000000041000005 | ...           | ...
                                ...
0000000041005000 | FF15 00000042 | call qword ptr [42000000]
0000000041005006 | ...           | ...
                                ...
0000000042000000 | 01100041      | ...
0000000042000004 | 00000000      | ...
```

Теперь рассмотрим более сложную функцию:

```
int __cdecl Bar(char *str, int size) { /* код функции */ }
```

Функция *Bar* принимает два аргумента и возвращает 32-битный integer. Вызов такой функции требует размещения на стеке значений *str* и *size* в обратном порядке (справа налево) и возвращения стека в изначальное положение после вызова в соответствии с соглашением о вызове «*cdecl*»[2]. Допустим, что адрес Bar тот же — 0x41001001. Вызов такой функции будет выглядеть следующим образом:

```
  Адрес   | Код операции | Мнемоника инструкции
40FFFFFE | 50           | push eax ; size
40FFFFFF | 51           | push ecx ; str
41000000 | E8 FC0F0000  | call 41001001
41000005 | 8945 FC      | mov dword ptr [ebp-4], eax
41000008 | 83C4 08      | add esp, 8
```

Перехват такой функции ничем не отличается от перехвата функции *Foo*, рассмотренной выше. Однако, наша функция, на которую требуется заменить *Bar*, должна следовать тому же соглашению о вызове, что и *Bar*. В данном случае «*cdecl*». Это необходимо для того, чтобы компилятор сгенерировал совместимый с вызывающей функцией код, то есть, чтобы наша функция “знала”, где какой аргумент находится на стеке и нужно ли перед возвращением “отчищать” стек от аргументов.

В случае с 64-разрядным приложением, в ABI (Application Binary Interface), который реализован компилятором MSVC (Microsoft Visual C++), определено всего одно соглашение о вызове — «*fastcall*» (x64). Передача первых четырёх аргументов осуществляется через регистры *rcx*, *rdx*, *r8*, *r9* соответственно. В случае передачи чисел с плавающей точкой (дробных) используются SSE регистры *xmm0*, *xmm1*, *xmm2* и *xmm3*. Все дополнительные аргументы помещаются на стек [3].

Перехват в вызываемой функции в отличие от предыдущего метода, подразумевает перехват непосредственно внутри самой функции, вызов которой необходимо перехватить. Этот подход наиболее часто используется, так как позволяет перехватить сразу все точки вызова функции. Исключением здесь являются встроенные функции (inline functions). Компилятор может оптимизировать вызов функции путём встраивания тела функции в те места, из которых осуществляется вызов. Таким образом, вызываемая функция становится частью вызывающей и фактически вызова функции не происходит.

Перехват осуществляется путём перезаписи инструкций по адресу вызываемой функции, что подразумевает наличие дизассемблера, если требуется автоматизация этого процесса. Так как код оригинальной функции изменяется, возникает вопрос: каким образом осуществить вызов оригинальной функции. Наиболее удачное решение было предложено Microsoft Research, реализованное в библиотеке Detours [1]. Их подход заключается в сохранении перезаписываемых инструкций в отдельную область памяти с добавлением к ним инструкции безусловного перехода обратно в перехватываемую функцию. Получившийся набор инструкций называется *функцией-трамплином*, который может быть использован для вызова оригинальной функции. Функция подставляемая вместо исходной называется *функцией-петлёй* (detour function).

Ход выполнения до перехвата:

```
[A] -> [B] -> [A]
```

Здесь A — функция вызывающая B, а B — вызываемая функция, которую планируется перехватить. Управление программой принадлежит функции A, затем она вызывает функцию B и контроль передается функции B, которая после завершения работы, передаёт управление обратно функции A.

Ход выполнения после перехвата:

```
[A] -> [C] -> [D] -> [B] -> [C] -> [A]
```

Здесь C — функция-петля, а D — функция-трамплин. Функция A вызывает функцию B, однако вместо этого происходит вызов функции C, которая через функцию-трамплин D осуществляет вызов оригинальной функции B. После завершения работы, функция B передаёт управление обратно функции C напрямую, не возвращаясь в функцию D, и, наконец, функция C возвращает управление функции A.

Такой подход не только решает проблему вызова оригинальной функции, но также даёт возможность контролировать момент вызова, который, к тому же, является необязательным. Функция-петля может не вызывать оригинальной функции, а сразу возвращать управление. Это возможно благодаря сохранению перехватываемой функции через использование функции-трамплина. Иными словами, перехватываемая функция изымается из потока выполнения программы и для неё создаётся новая точка входа — функция-трамплин, которую можно использовать для вызова.

Рассмотрим на примере схему перехвата. Предположим, что перехватываемая функция имеет следующую сигнатуру и тело:

```
int Func_B(int value) { return (value * 2); }
```

В скомпилированном виде эта функция будет иметь следующий вид (оптимизации отключены для наглядности):

```
  Адрес   | Код операции | Мнемоника инструкции
00000000 | 55           | push ebp
00000001 | 8BEC         | mov ebp, esp
00000003 | 8B45 08      | mov eax, dword ptr [ebp+8]
00000006 | D1E0         | shl eax, 1
00000008 | 5D           | pop ebp
00000009 | C3           | ret
```

Для перехвата в 32-разрядном приложении потребуется перезаписать не более 6 байт поверх первых трёх инструкций:

```
  Адрес   | Код операции | Мнемоника инструкции
00000000 | E9 FBFFFF3F  | jmp 40000000
00000005 | 90           | nop
00000006 | D1E0         | shl eax, 1
00000008 | 5D           | pop ebp
00000009 | C3           | ret
```

Таким образом, при вызове функции *Func_B* произойдёт передача управления функции по адресу 0x40000000. Для правильного отображения кода в отладчике, оставшиеся от оригинальных инструкций байты заменяются инструкцией *nop* (на работу программы это никак не влияет). Функция-трамплин имеет следующий вид:

```
  Адрес   | Код операции | Мнемоника инструкции
41000000 | 55           | push ebp
41000001 | 8BEC         | mov ebp, esp
41000003 | 8B45 08      | mov eax, dword ptr [ebp+8]
41000006 | E9 FBFFFFBE  | jmp 00000006
4100000B | ...          | ...
```

При условии, что функция-петля реализует соглашение о вызове, используемое перехваченной функцией (в данном случае *Func_B*), и вызов *Func_B* осуществляется через функцию-трамплин, обе функции можно использовать как обычно, не внося дополнительных изменений.

В случаях, когда в функцию-трамплин попадают инструкции относительного перехода, их адреса переписываются. Если значение расстояния не вмещается в диапазон относительного перехода, то такой переход заменяется абсолютным.

Стоит отметить пару моментов. Если приложение 64-разрядное, в худшем случае потребуется перезаписать 12 байт, в остальном также как и с 32-разрядным. Если вся перехватываемая функция имеет длину меньше, чем необходимо для записи перехвата, то потребуется переход на ближайшее свободное место (описано в первом методе). Часто после функции имеется отступ с целью выравнивания на 16-байтной границе, который можно использовать при нехватке места. Также возможна ситуация, когда перехватываемая функция поддерживает перехват. В таком случае первая инструкция предназначена для записи относительного перехода, а перед самой функцией имеется место для записи абсолютного перехода.

Перехват через таблицу виртуальных функций осуществляется путём замены адреса перехватываемой функции в таблице виртуальных функций. Такой метод перехвата специфичен для программ, написанных на языках объектно-ориентированного программирования (ООП). Для каждого объекта, у которого имеется хотя бы одна виртуальная функция, создаётся таблица, в которой хранятся адреса функций, реализующих виртуальные функции. Вызов происходит путём получения адреса функции из этой таблицы:

```
  Адрес   | Код операции | Мнемоника инструкции
00000018 | 8B10         | mov edx,dword ptr [eax]
0000001A | 8B52 08      | mov edx,dword ptr [edx+8]
0000001D | 8BC8         | mov ecx,eax
0000001F | FFD2         | call edx
00000021 | ...          | ...
```

Такой вызов называется *непрямым* (indirect call) и легко может быть перенаправлен. При непрямом вызове через таблицу виртуальных функций, как правило, используется соглашение о вызове «*thiscall*», но также могут использоваться и другие, например «*stdcall*».

В некоторых языках используется понятие «абстрактный метод», значение которого совпадает с понятием «чистая виртуальная функция» в C++. Такие функции тоже реализуются через таблицу виртуальных функций и в этом плане ничем не отличаются от обычных виртуальных функций.

В 64-разрядных приложениях отличий нет.

Перехват через таблицу импорта подразумевает замену адреса в таблице импортированных функций. Любое приложение, так или иначе, использует функции из сторонних модулей. *Модулем* называют исполняемый файл, которым является само приложение (EXE-файл), а также загружаемые библиотеки (DLL-файлы). В файле модуля хранится таблица импорта — список сторонних модулей и их функций, использованных исполняемым файлом.

При загрузке исполняемого файла в память, загрузчик считывает этот список и подгружает необходимые модули в виртуальное адресное пространство процесса приложения. После чего, осуществляется запись адресов указанных в списке функций в эту таблицу. Функции, загруженные таким образом, называются *импортированными*. Импортированные функции, как и виртуальные, вызываются через непрямой вызов:

```
  Адрес   | Код операции   | Мнемоника инструкции
009ACB39 | FF15 9437F900 | call dword ptr [00F93794]
009ACB3F | ...           | ...
                              ...
00F93794 | 64E710C8      | ...
```

В данном примере для перехвата достаточно изменить адрес функции по адресу 0x00F93794. В 64-разрядном приложении перехват осуществляется аналогично.

Логично предположить, что этот метод, в отличие от трёх других, используется только при динамическом подходе к перехвату, так как для его реализации требуется загрузка исполняемых файлов в оперативную память.

При выборе метода перехвата функции необходимо исходить из специфики вызывающего кода и поставленной цели. Универсальным является метод перехвата в вызываемой функции, который при хорошей реализации будет работать безотказно. Однако не следует бездумно применять его каждый раз, когда необходимо перехватить функцию, всегда существует вероятность наличия более простого решения.

Библиографический список

1. Galen Hunt, Doug Brubacher. Detours: Binary Interception of Win32 Functions // Third USENIX Windows NT Symposium. — July 1999. — P.1-9.

2. Cdecl. Microsoft Docs [Электронный ресурс]. — Режим доступа: https://docs.microsoft.com/en-us/cpp/cpp/cdecl (дата обращения 15.02.2020 г.)

3. x64 calling convention. Microsoft Docs [Электронный ресурс]. — Режим доступа: https://docs.microsoft.com/en-us/cpp/build/x64-calling-convention (дата обращения 15.02.2020 г.)

УДК 519.688; ГРНТИ 20.53

ПРОЦЕСС ЗАГРУЗКИ ИСПОЛНЯЕМЫХ ФАЙЛОВ В ОПЕРАЦИОННОЙ СИСТЕМЕ WINDOWS

Д.Н. Егоров

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, dmitryegorov48@gmail.com

Аннотация. В статье описан процесс загрузки исполняемых файлов в ОС Windows. Рассматриваются различные стадии создания процесса и подготовки к исполнению загружаемых программ. Приведен список задач загрузчика исполняемых файлов.

Ключевые слова: исполняемый файл, операционная система, native-приложение, патч, загрузчик.

WINDOWS IMAGE LOADING PROCESS

D.N. Egorov

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, dmitryegorov48@gmail.com

The summary. Image loading process in Windows operating system is characterized. Process creation stages are described. Image loader responsibilities are specified.

Keywords: image file, operation system, native-application, patch, image loader.

Под исполняемым файлом понимается файл, содержащий исполняемый код, который загружается операционной системой для последующего выполнения. Исполняемый код — это последовательность инструкций реального процессора, в случае, если программа является native-приложением, или виртуального, так называемый промежуточный код или байт-код.

Помимо кода в исполняемом файле, как правило, хранятся и другие данные, необходимые для работы программы и описывающие её характеристики. Например, информация о том, набор инструкций какой архитектуры процессора используется программой (x86, ARM, MIPS, и т.д.), разрядность (8, 16, 32 или 64-бита), адрес начала исполнения (точка входа) и другие. Для того чтобы иметь представление где какая информация находится в исполняемом файле, используется строго обозначенная разметка — формат файла.

В операционной системе Windows к исполняемым файлам относят:

1. Windows native-приложения.
2. .NET приложения.
3. DOS .bat и .cmd файлы.
4. DOS приложения.
5. 16-битные Windows приложения.

К Windows native-приложениям относятся программы, скомпилированные в набор инструкций реального процессора, используемого электронной вычислительной машиной (ЭВМ), и использующие подсистему Windows.

Приложения .NET отличаются от Windows native-приложений только тем, что в качестве набора инструкций используют промежуточный код Intermediate Language (IL) и запускаются внутри процесса Common Language Runtime (CLR).

Файлы .bat и .cmd являются программами, интерпретируемыми командной строкой DOS и её эмулятором в Windows. Являются текстовыми файлами и представляют собой последовательность запросов с поддержкой операторов разветвления, таких как условия, циклы и безусловные переходы.

DOS и 16-битные Windows приложения — это приложения, написанные под операционные системы DOS и старые версии Windows, такие как Windows NT 3.1. Особенностью таких приложений является то, что они не могут быть запущены в 32 и 64-битных версиях Windows напрямую.

Процесс загрузки исполняемого файла начинается с создания такого объекта внутри операционной системы как процесса, внутри которого приложение исполняется. Создание процесса состоит из нескольких стадий:

1. Проверка входных параметров и их преобразование в аналогичные структуры, использующиеся внутри операционной системы.
2. Открытие исполняемого файла.
3. Создание объекта исполняемого процесса.
4. Создание и инициализация основного потока.
5. Инициализация используемой подсистемы.
6. Запуск основного потока (если во входных параметрах не указано иное).
7. Завершение инициализации виртуального адресного пространства (ВАП) в контексте созданного процесса и потока.
8. Начало исполнения в точке входа.

Стоит заметить, что приведенный список стадий создания объекта «процесс» описывает создание нового процесса из уже существующего, родительского процесса. На момент загрузки операционной системы уже создано несколько основных процессов, алгоритм создания которых отличается. Рассмотрим каждую из стадий подробно.

Стадия 1. Проверка и преобразование входных параметров.

На этой стадии, как следует из названия, происходит проверка входных параметров, таких как приоритет процесса, флаг использования отладчика и других. В зависимости от этого устанавливаются значения внутренних структур ОС, которые будут использованы в следующих стадиях.

Стадия 2. Открытие исполняемого файла.

Здесь происходит переход из user-mode в kernel-mode и осуществляется повторная проверка входных параметров из соображений безопасности. Также выполняется определение типа исполняемого файла и в зависимости от этого выбирается, требуется ли для запуска другой исполняемый файл, и если да, то какой. Помимо этого, в случае необходимости, происходит проверка цифровых подписей.

Стадия 3. Создание объекта исполняемого процесса.

На этой стадии создаётся объект исполняемого процесса, для чего [1]:

1. Инициализируется объект типа EPROCESS.
2. Создаётся виртуальное адресное пространство (ВАП).
3. Инициализируется структура KPROCESS.
4. Инициализируется ВАП.
5. Инициализируется Process Environment Block (PEB).
6. Завершается создание исполняемого процесса.

Стадия 4. Создание и инициализация основного потока.

После создания объекта исполняемого процесса происходит создание первого основного потока. Процесс создания потока включает в себя:

1. Создание и инициализацию объекта исполняемого потока.
2. Запись времени создания потока.
3. Инициализацию Thread Environment Block (TEB).
4. Инициализацию структур ETHREAD и KTHREAD.

Стадия 5. Инициализация используемой подсистемы.

На этой стадии, в случае успешности выполнения предыдущих стадий, происходит инициализация определенной ранее используемой подсистемы, путём передачи ей управления.

Стадия 6. Запуск основного потока.

К этому моменту была произведена необходимая подготовительная работа для запуска основного потока, в котором завершится создание процесса и начнётся исполнение программы. На этом этапе определяется, запускать ли этот процесс.

Стадия 7. Завершение инициализации ВАП.

Здесь выполняется отправка отладочных уведомлений о создании процесса, осуществляется переход из kernel-mode обратно в user-mode, осуществляется вызов загрузчика PE32-файлов, происходит загрузка необходимых DLL файлов и вызов их точек входа с параметром DLL_PROCESS_ATTACH [2].

Стадия 8. Начало исполнения в точке входа.

На последней стадии происходит передача управления внутри потока на точку входа в приложение и с этого момента начинается исполнение непосредственно загруженного исполняемого файла.

Загрузчик работает в user-mode режиме и находится в файле *ntdll.dll*. Его работа сводится к нескольким основным задачам:

1. Инициализация состояния приложения.
2. Считывание таблиц импорта и экспорта.
3. Загрузка и выгрузка DLL-файлов.
4. Обработка манифест-файлов.
5. Поиск патчей совместимости.
6. Перенаправление API.

7. Осуществление обратной совместимости.

Первая задача включает в себя инициализацию кучи и настройку локальных хранилищ для потока (TLS) и волокна (FLS).

Считывание таблиц импорта и экспорта осуществляется с целью выявления необходимых для загрузки DLL-файлов. Загрузчик считывает таблицу импорта загружаемого исполняемого файла и для каждой записи (DLL-файла) осуществляет считывание таблицы экспорта с целью проверки наличия нужных записей. Таким образом, перед загрузкой происходит сопоставление записей для каждого из DLL-файлов в таблице импорта загружаемого исполняемого файла с записями в таблице экспорта требуемого для загрузки DLL-файла. В таблице экспорта также может содержаться специальная запись, которая перенаправляет загрузчик в другой DLL-файл.

После проверки наличия записей из таблицы импорта загружаемого исполняемого файла в таблице экспорта DLL-файла, происходит загрузка последнего в ВАП созданного процесса. Помимо загрузки непосредственно в момент инициализации, загрузчик также ответственен за загрузку и выгрузку во время исполнения приложения. Таким образом, приложение может использовать DLL-файлы, не указанные в таблице импорта. Это полезно, когда наличие определенной версии библиотеки заранее неизвестно и необходимо осуществить перебор. Например, библиотека XInput имеет несколько различных версий, некоторые из которых могут присутствовать в системе, а другие — нет [3].

Манифест-файлы требуются для поддержки таких систем ОС Windows как Side-by-Side (SxS) и Multiple Language User Interface (MUI), а также для описания ресурсов, находящихся в файле (например, изображения иконок, курсоров и разметка диалоговых окон).

Поиск патчей совместимости включает в себя проверку наличия приложения в базе данных совместимости и загрузку соответствующих исправлений для поддержания нормальной работы приложения. Когда приложение падает, Windows выводит окно с предложением поиска существующего решения. При выборе этого пункта будет осуществлен поиск патчей для приложения в удаленной базе данных совместимости. Так как приложение было запущено, в локальной базе нужных записей обнаружено не было.

Перенаправление API необходимо для обеспечения работы так называемых Universal Windows Platform (UWP) приложений, использующих единый API. Его реализация зависит от конкретной аппаратной платформы, на которой запущена ОС.

Осуществление обратной совместимости отличается от поиска патчей в базе совместимости тем, что вместо изменения поведения самого приложения, изменяется поведение ОС через механизм под названием SwitchBack [1]. Таким образом, ОС может имитировать работу предыдущих версий. Этот механизм активируется при выставлении пункта «обратной совместимости» в свойствах исполняемого файла на вкладке «совместимость».

Библиографический список

1. Yosifovich P., Ionescu A., Russinovich M.E., Solomon D.A. – Windows Internals, Part 1, 7th Edition – 2017.
2. Russinovich, Solomon, Ionescu - Windows Internals (5th edition) – 2009.
3. XInput Versions [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/en-us/windows/win32/xinput/xinput-versions (дата обращения 17.02.2020)

УДК 004.42; ГРНТИ 50.50

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ПРОВЕРКИ ЗНАНИЙ С ИСПОЛЬЗОВАНИЕМ КОМПЕТЕНЦИЙ

Н.Н. Степанов
Рязанский государственный радиотехнический университет,
Российская Федерация, Рязань, nikita.stepanov150@gmail.com

Аннотация. В работе рассматривается разработка приложения проверки знаний, причины его использования и возможная модель.
Ключевые слова: компетенция, веб-приложение, проверка знаний.

DEVELOPMENT OF A KNOWLEDGE VERIFICATION WEB APPLICATION BY USING COMPETENCES

N.N. Stepanov
Ryazan State Radio Engineering University,
Russia, Ryazan, niktia.stepanov150@gmail.com

The summary. The paper discusses the development of a knowledge application, the reasons for its use and a possible model.
Keywords: competence, web-application, quiz.

В наше время новые технологии проникают во все сферы деятельности. В сфере информационных технологий, чтобы оставаться востребованным специалистом на рынке, необходимо постоянно изучать что-то новое и совершенствовать свои навыки, так как тенденции имеют свойство изменяться каждый месяц. Для языков программирования стало уже нормой выпускать мажорные обновления каждые 3-6 месяцев, которые привносят что-то новое и полезное. Одно из таких обновлений может кардинально изменить существующий подход к разработке. Популяризация большого количества фреймворков (фреймворк – программное обеспечение, облегчающее разработку, также может определять архитектуру программной системы) на той или иной платформе.

В настоящее время недостаточно знать только какой-либо язык программирования. Необходимо также войти во все современные общепринятые стандарты отрасли, коих за почти 15 лет накопилось немало. Стать инженером, который может правильно анализировать ситуации и принимать дальновидные решения. Сложность технологий постоянно увеличивается. Не всегда можно самостоятельно разобрать каждый аспект. Порой необходимо рассмотреть использование того или иного функционала с разных сторон, приняв во внимание все их плюсы и минусы. Не бывает идеальных ситуаций. Поэтому высококвалифицированные специалисты обязаны разбираться во всех имеющихся стандартах, иметь ввиду все аспекты и тонкости современной разработки. Из-за высоко поднятых требований к специалистам процесс обучения новых разработчиков уже должен учитывать современные требования к знаниям, также необходимо идти в ногу со временем, постоянно изменяя курс.

В вузах существует возможность использования компетенций, которые помогают понять к чему именно готовят студентов, какой спектр задач они могут решать, какие технологии они освоили, с какими трудностями они могут столкнуться.

Проблема возникает в том, что в данный момент очень сложно оценить каждого студента – к нему нужен индивидуальный подход преподавателя: необходимо проверить ответы; присвоить заданиям компетенции; выставить баллы за задание; вычислить полученную оценку в баллах. К тому же в программировании не существует однозначно правильного и неправильного решения. Та же самая задача вполне может решиться миллионом способов. Высококвалифицированный специалист должен уметь принять во внимание все достоинства и недостатки выбранной архитектуры и как следует проработать ее использование. Для этого необходимо качественно анализировать каждое предложенное решение. Такой пристальный

подход занимает очень много времени, хотя и дает положительные результаты. И необходимо понимать, что только тестирование разработчиков программного обеспечения – крайне неверный способ судить о его квалификации. Для этого лучше всего также давать тестовое задание и смотреть как он может справляться с возникшими трудностями и поставленными задачами: знает ли он где можно самостоятельно найти ответ на вопрос и решить задачу, умеет ли пользоваться документацией по языкам программирования, насколько хорошо ему известна платформа, под которую он разрабатывает приложения, насколько хорошо он владеет сопутствующими инструментами, которые уже стали стандартом. Все это накладывает ограничения на преподавателя и его рабочее время.

Хорошо подходит для этого автоматизированные системы непрерывной интеграции. Когда студент загружает задание, она автоматически осуществляет проверку задания на соблюдение общепринятых правил оформления кода, поиск типичных слабых мест. Имеется возможность написать собственное правило анализа и построения приоритетов. Такие системы общеприняты и стали стандартом разработки. На основе отчетов от системы анализа качества кода можно судить о качестве работы. Эти системы позволяют не заниматься преподавателю рутиной работой - искать типичные ошибки, например:

- название класса с маленькой буквы;
- название переменной с большой буквы;
- имя метода с большой буквы;
- неверный регистр для констант;
- неинформативные имена переменных и компонентов;
- неверное форматирование;
- типичные неверные примеры использования исключений;
- статическая проверка на возможные исключения;
- возможное наличие копирайтов или лицензии;
- какие-либо другие особенности конкретных языков программирования.

Разрабатываемое веб-приложение является системой, которая агрегирует в себе все качества, как для промышленной разработки, так и для проверки знаний студентов. Оно снизит нагрузку преподавателя на тех задачах, которые можно автоматизировать. Тогда преподаватель будет заниматься работой в качестве эксперта – необходимо проставить компетенции по каждым курсам или тестам, а приложение в автоматическом режиме оценит их. Что увеличивает производительность и качество работы. Также довольно полезным является факт возможности накопления статистики, появляется возможность исследовать зависимости от того или иного подхода к преподаванию. Например, решили изменить курс, после этого произвели тестирование, получили статистику. Это открывает возможности к дополнительным исследованиям.

Целью данной работы является разработка веб-приложения проверки знаний с использованием и оценивания его компетециии:

- использовать набор курсов в качестве исходных данных;
- преподаватель выставляет компетенции заданиям;
- работодатель может просмотреть компетенции каждого студента;
- хранить историю решения задач, в том числе и неудачные попытки;
- провести тестирование студента в соответствии с курсом и заданиями;
- уведомлять преподавателя о статусе прохождения курса студентом;
- выдать результат тестирования;
- иметь графический интерфейс;
- иметь возможность загрузки оценок из внешних систем;
- необходимо иметь возможность интегрироваться со всеми известными системами для выгрузки каких-либо данных;
- высокую отказоустойчивость;

- возможность следить за активностью пользователя;
- вести аудит событий;
- простая навигация по имеющемуся материалу;
- вести статистику прохождения каждого задания в отдельности;
- легкую расширяемость.

Веб-приложение должно иметь следующую ролевую модель:

- студент – решает задания, проходит определенные курсы;
- преподаватель – составляет индивидуальный план обучения, присваивает заданиям компетенции, выгружает курсы из внешних систем, загружает оценки из других систем оценки компетенций;
- работодатель – просматривает компетенции студента, участвует в составлении курсов с точки зрения необходимых практических навыков.

Главной сущностью является курс. Имеется возможность динамически создавать любой из объектов внутри системы. Курс состоит из нескольких заданий. Каждое задание может быть с выбором ответа. Задание включает в себя следующие составляющие:

- формулировка;
- теоретический материал;
- пример выполнения задания.

Веб-приложение будет разработано на языке Java [1] и JavaScript [2]. В качестве основной архитектуры разработки был выбран микросервисный подход [3]. Архитектурный стиль микросервисов — это подход, при котором единое приложение строится как набор небольших сервисов, каждый из которых работает в собственном пространстве и взаимодействует с остальными, используя различные механизмы, например HTTP или Message Queue (очереди сообщений) [4]. Эти сервисы построены вокруг потребностей и развертываются независимо с использованием полностью автоматизированной среды. Например, довольно популярным решением является kubernetes от Google [5] – оркестратор контейнеров с открытым исходным кодом для автоматизирования развертывания приложений, с возможностью быстрого горизонтального масштабирования, управления, сбора различных метрик. Существует минимум централизованного управления этими сервисами. Сами по себе эти сервисы могут быть написаны на разных языках и использовать разные технологии хранения данных.

Приложение будет разработано в рамках выпускной квалификационной работы магистранта. Планируется внедрение веб-приложения в учебный процесс [6].

Библиографический список

1. Блинов И.Н., Романчик В.С. Java. Методы программирования: учеб.-метод. пособие. – Минск: Изд-во «Четыре четверти», 2013. – 896 с.
2. David Herman. Effective JavaScript. Addison-Wesley, 2011. – 950 с.
3. Eberhard Wolff. Microservices: Flexible Software Architecture. 2016. – 1026 с.
4. Gwen Shapira, Todd Palino. Kafka: The Definitive Guide: Real-Time Data and Stream Processing at Scale, O'Reilly Media, Inc, 2017. - 836 с.
5. Kelsey Hightower, Brendan Burns, Joe Beda.Kubernetes: Up and Running: Dive Into the Future of Infrastructure. 2017. – 650 с.
6. Пруцков А.В. Особенности преподавания промышленной разработки программных продуктов в технических вузах // В сборнике: Современные технологии в науке и образовании - СТНО-2017 сборник трудов II Междунар. науч.-техн. и науч.-метод. конференции: в 8 т. Ряз. гос. радиотехн. ун-т. 2017 - С. 35-36.

УДК 338.45:004.03; ГРНТИ 45.01.85

ПРОБЛЕМА ЦИФРОВИЗАЦИИ В ОБЛАСТИ ЭЛЕКТРОЭНЕРГЕТИКИ

А.Ю. Громов, Д.Г. Шмелева

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань gromov.a.y@evm.rsreu.ru, tearstotiara@mail.ru

Аннотация. В работе рассматриваются проблемы при внедрении цифровизации в области электроэнергетики. Рассмотрена используемая за рубежом технология интеллектуальных сетей.

Ключевые слова: концепция цифровизации, интеллектуальные сети.

THE PROBLEM OF DIGITALIZATION IN THE ELECTRICPOWER INDUSTRY

A.Yu. Gromov, D.G. Shmeleva

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, gromov.a.y@evm.rsreu.ru, tearstotiara@mail.ru

The summary. The paper discusses the problems of digitalization in the electricpower industry in Russia. The technology of smart greed used abroad is considered.

Keywords: the concept of digitalization, smart greed.

Приоритетным направлением в обеспечении оказания услуг транспорта электроэнергии является обеспечение бесперебойного поступления электроэнергии и быстрого подключения новых потребителей. Применение информационных технологий в данной области, позволяет повысить качество работы, исключить ошибки из-за невнимательности (человеческий фактор), а также позволяет значительно сократить время, затраченное на получение результатов запроса работника и поиск нужной информации.

В настоящее время разработан и подписан федеральный закон «О внесении изменений в отдельные законодательные акты Российской Федерации в связи с развитием систем учета электрической энергии (мощности) в Российской Федерации», вводящий единые требования к интеллектуальным приборам и системам учета электрической энергии. Закон содержит информацию о переходе от обычных счетчиков к «умным» до 1 января 2022 г [1].

Из вышесказанного можно сделать вывод, что для осуществления цифровизации необходима система, которая сможет обслуживать умные счетчики, а также прогнозировать нагрузку на сети и эффективно распределять ресурсы. Цифровизация в электроэнергетике – это оцифровка различных технологических процессов производства, потребления и распределения электроэнергии, а также технологии управления этими процессами. Это переход от ситуативного решения возникающих проблем различного уровня к полноценному анализу и мониторингу энергосистемы. Однако такой емкий процесс не может происходить быстро.

Главное препятствие внедрению цифровизации в энергетике на сегодняшний день – это отсутствие единых стандартов и требований необходимых для сбора, хранения и обработки большого объема данных. Помимо этого, важной проблемой остается переход от старого оборудования (которым пользуется большинство компаний) к более современному, так как интеграция устаревшего оборудования с высокотехнологичными секторами обойдется дороже, чем его замена.

Не менее важными проблемами являются отсутствие достаточного финансирования и отсутствие координации между компаниями-гигантами – зачастую они внедряют новые технологии без учета требований электроэнергетического рынка и потребностей остальных его участников. Также проблемой является отсутствие практического опыта – нехватка квалификации приводит к невысокой производительности труда и выбору неверных технологических и управленческих решений. Важно уделить внимание кибербезопасности – необходимо защитить энергообъкеты и данные от кибератак.

Помимо сложностей при внедрении цифровизации проблемы начинаются на начальном этапе: оценке реальной эффективности цифровизации электроэнергетики. Основные

причины этому – это изменения, связанные с внедрением цифровизации, структуры сети и первичных элементов, изменения правил и стандартов, а также применение различных поколений технологий.

Цифровизация приведет к снижению потерь, инвестиционных затрат на покупку основных фондов (CAPEX), операционные расходы (OPEX), сократит перерывы в электроснабжении из-за поломки оборудования (SAIDI) и увеличит доступность трансформаторных подстанций. Однако эффекты модели достигаются только в результате параллельной работы всех функциональных областей [2].

Наиболее эффективно проводить в 3 этапа, при этом распределить части технологий по разным этапам, чтобы показатели эффективности пропорционально увеличивались. Единая цифровая среда данных предоставит возможность проведения аналитических исследований для принятия оптимального решения в той или иной области. Также позволит производить анализ состояния оборудования, вести прогнозы и последствия отказов, снижать риски выхода из строя оборудования.

Концепция цифровизации в области электроэнергетике приведена на рисунке 1.

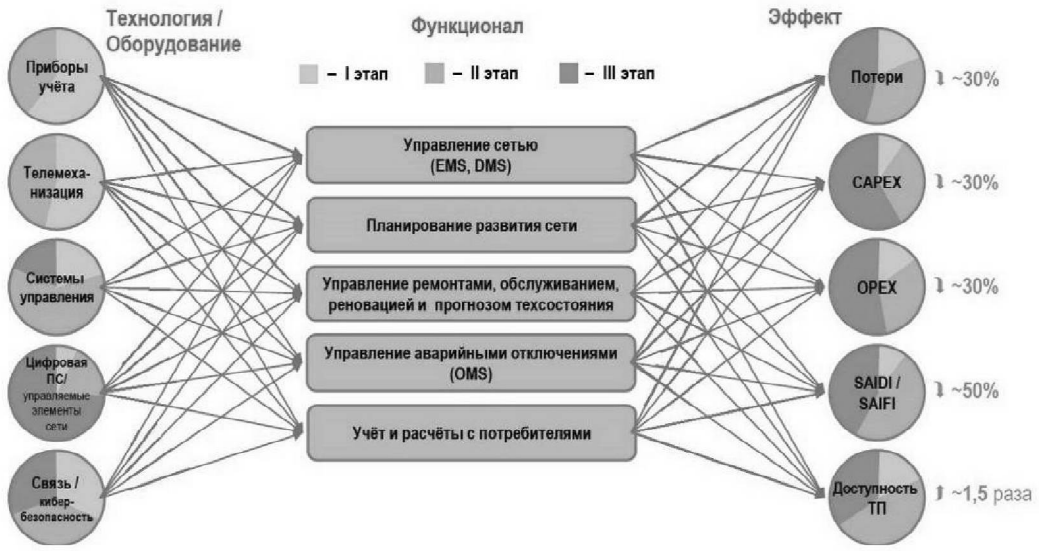


Рис. 3. Концепция цифровизации

Во всем мире набирает популярность технология «Интеллектуальной сети» (Smart Grid). IEEE определяет Smart Grid как концепцию полностью интегрированной, саморегулирующейся и самовосстанавливающейся электроэнергетической системы, имеющей сетевую топологию и включающей в себя все генерирующие источники, магистральные и распределительные сети и все виды потребителей электрической энергии, управляемые единой сетью информационно-управляющих устройств и систем в режиме реального времени. Помимо этого такие сети имеют дело с большим объемом данных, которые нужно постоянно хранить, накаливать и обрабатывать, что еще больше усложняет их построение.

Сейчас существуют несколько зарубежных проектов, которые используют данную технологию, например, проект «FENIX» (Flexible Electricity Networks to Integrate the expected Energy Evolution). Проект построения гибкой электрической сети (Испания и Великобритания) или проект «ADDRESS» (Active Distribution network with full integration of Demand and distributed energy RESourceS). На данный момент в России только начинают появляться такого рода системы.

Архитектура таких сетей состоят из следующих элементов: интеллектуальные датчики и сенсоры для распределенных сетей; системы, занимающиеся управлением и контролем решений в области автоматизизации; технологические решения, которые используются в распределенных сетях; интеллектуальные счетчики в виде программно-аппаратных средств [3].

Структура интелекктуальных сетей представлена на рисунке 2.

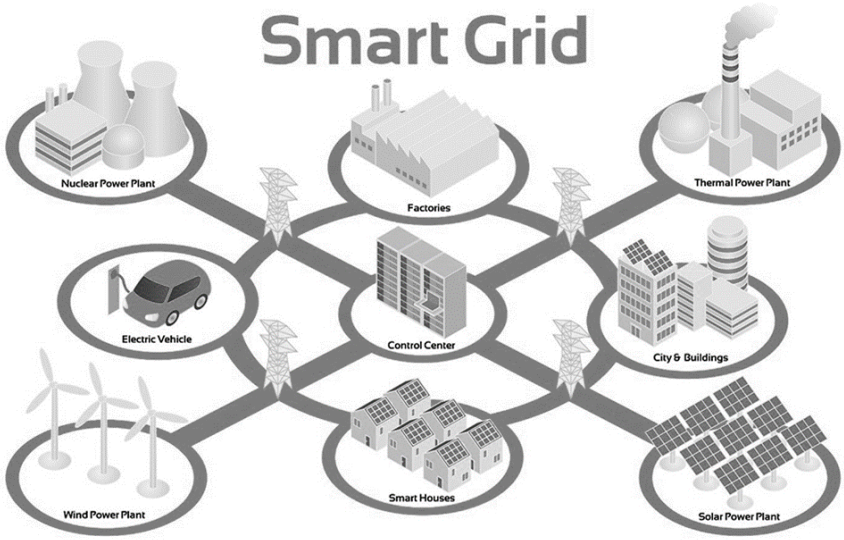


Рис. 4. Структура интеллектуальных сетей

Сейчас в России нет возможности подключения умных сетей из-за сложности их архитектуры. Во-первых, это связано с оборудованием, используемом при подключении. Изменение обычных счетчиков на умные довольно затратный процесс. Во-вторых, отечественные умные счетчики отличаются от счетчиков, использованных за рубежом. Сейчас счетчики могут автоматически передавать данные в систему, но отслеживание краж и поломок производятся специалистами вручную. Также разработанные системы (Smart Grid Integrated Communications) не подходят для работы в России из-за различий в законодательстве и оборудовании компаний. Помимо этого, стоит учитывать риски при заимствовании интеллектуальных систем. При введении санкций права владения такими системами могут меняться, что повлечет к отключению пользователей и обвалу электросетевого рынка. Именно поэтому стоит вопрос в разработке собственных уникальных систем и внедрению отечественного оборудования.

Таким образом, можно сделать вывод, что несмотря на то, что в зарубежных системах реализована часть необходимых функций, с их помощью невозможно провести полноценный анализ полученных данных для дальнейшего планирования и эффективного практического использования. Следовательно, для успешного осуществления цифровизации предприятия и области электроэнергетики в целом, а также преодоления связанных с этим проблем, необходима разработка новых программных решений и систем.

Библиографический список

1. ФЗ «О внесении изменений в отдельные законодательные акты Российской Федерации в связи с развитием систем учета электрической энергии (мощности) в Российской Федерации»;

2. Усманова Т. Х., Исаков Д. А. Новая парадигма в модернизации системы электроэнергетики // Большая Евразия: Развитие, безопасность, сотрудничество. 2019. №2-1;

3. Amam Hossain Bagdadee, Li Zhang Smart Grid: A Brief Assessment of the Smart Grid Technologies for Modern Power System // Journal of Engineering Technology Volume 8, Issue 1, Jan. 2019, PP.122-142.

УДК 004.9; ГРНТИ 20.53.19

АЛГОРИТМЫ, ИСПОЛЬЗУЕМЫЕ В РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМАХ МЕДИА-ПОРТАЛОВ И ИНТЕРНЕТ-ТОРГОВЛИ

А.Т. Кежватова, А.Ю. Громов
Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, kejwatov.al@yandex.ru

Аннотация. В работе рассматриваются существующие алгоритмы формирования рекомендаций, разработанные известными крупными компаниями и используемые в собственных рекомендательных системах.
Ключевые слова: рекомендательные системы, CineMatch, DSSTNE, CatBoost.

ALGORITHMS USED IN THE RECOMMENDED SYSTEMS OF MEDIA PORTALS AND ONLINE TRADE

A.T. Kezhvatova, A.Yu. Gromov
Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, kejwatov.al@yandex.ru

The summary. The paper considers the existing algorithms for generating recommendations developed by well-known large companies and used in their recommendation systems.
Keywords: recommendation systems, CineMatch, DSSTNE, CatBoost.

В настоящее время крупные компании стараются использовать эффективные рекомендательные системы, чтобы быть конкурентоспособными на рынке. Алгоритмы, которые используются в рекомендательных системах, предполагают анализ большого количества данных. На этапе анализа данных возникают такие проблемы, как отсутствие масштабируемости, смещение результатов и снижение эффективности, отсутствие конфиденциальности и недостаточная защита персональных данных. В работе рассматриваются текущие тенденции развития алгоритмов, которые используются крупными компаниями в собственных рекомендательных системах.

1. Netflix – алгоритм CineMatch.

Веб-сайт Netflix формирует рекомендации автоматически с помощью алгоритма рекомендаций [1]. Алгоритм предусматривает обработку серверами Netflix информации из своих баз данных для того, чтобы определить, какие фильмы понравятся пользователю. Данный алгоритм учитывает следующие факторы:

– сами фильмы, которые объединены в группы по определенному признаку;

– рейтинги конкретного пользователя;

– совокупные рейтинги всех пользователей Netflix.

В качестве алгоритма рекомендаций Netflix использует алгоритм CineMatch. Алгоритм устанавливает соответствие между историей просмотра пользователя и рейтингом людей, которые имеют схожие истории. Он использует похожие профили для предсказания того, какие фильмы понравятся конкретному пользователю. Данные предсказания и являются рекомендациями. Они основаны на алгоритмах и статистике.

Прогноз формируется с сопоставления фильмов друг с другом, а не с сопоставления людей с фильмами, так как в базе гораздо меньше фильмов, чем пользователей Netflix.

Чтобы сделать прогноз, система:

– ищет в базе данных людей, оценивших один и тот же фильм;

– определяет, кто из этих же людей так же оценил второй фильм;

– вычисляет статистическую вероятность того, что людям, которым понравился первый фильм, так же понравится второй фильм;

– продолжает этот процесс, чтобы установить закономерность корреляций между рейтингами подписчиков различным фильмам.

Алгоритм успешно решает проблему масштабируемости. Сформированные с помощью алгоритма CineMatch рекомендации являются достаточно точными, но он принадлежит компании Netflix и применяется в рамках деятельности данной компании в области рекомендации фильмов.

2. Amazon – алгоритм DSSTNE.

На сайте amazon.com видно, что рекомендации отображаются под описанием продукта (рисунок 1). Некоторые из них генерируются из того, что приобрели другие клиенты, а некоторые – из истории браузера.

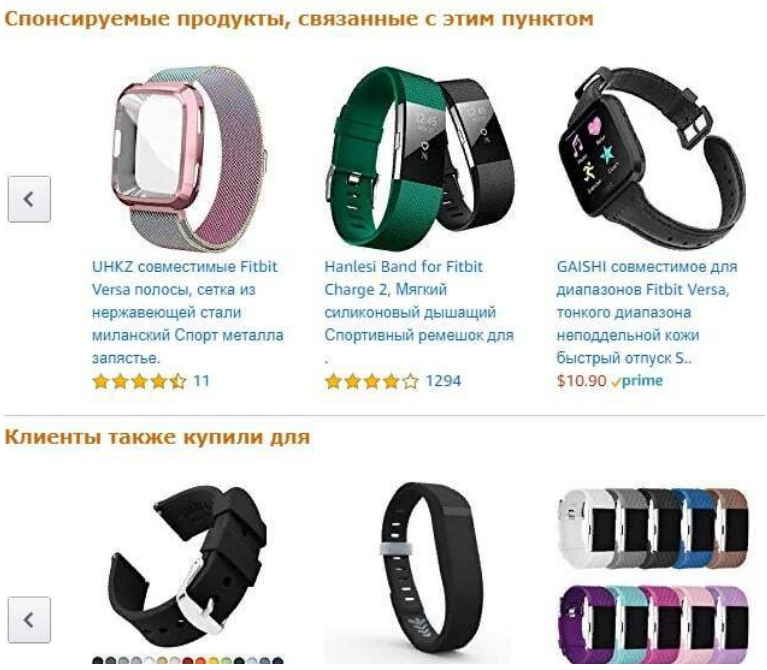


Рис. 1. Персональные рекомендации на Amazon

Персонализированные рекомендации не основаны на каком-либо предположении или догадке. Это алгоритм, который обрабатывает поведение пользователей в сети. В данный модуль могут подгружаться элементы, которые были больше всех просмотрены или уже кем-то приобретены.

Для формирования персональных рекомендаций компания Amazon использует алгоритм Deep Scalable Sparse Tensor Network Engine (DSSTNE).

DSSTNE – это библиотека программного обеспечения с открытым исходным кодом для обучения и развертывания моделей рекомендаций с разреженными входами, полностью подключенными скрытыми слоями и разреженными выходами [2].

DSSTNE был построен с рядом функций для производственных рекомендаций рабочих нагрузок [3]:

– Multi-GPU Scale: обучение и прогнозирование масштабируются, чтобы использовать несколько графических процессоров, распределяя вычисления и хранение параллельно модели для каждого слоя;

– Large Layers: модель-параллельное масштабирование позволяет использовать большие сети, чем это возможно с одним графическим процессором;

– Sparse Data: DSSTNE оптимизирован для быстрой производительности на разреженных наборах данных, распространенных в рекомендациях.

Данные функции позволяют избегать проблем с отсутствием масштабируемости, смещением результатов и снижением эффективности.

Представленный алгоритм используется в Amazon для создания персонализированных рекомендаций по продуктам непосредственно для клиентов в масштабе Amazon.

3. Яндекс – алгоритм CatBoost.

Компания Яндекс для решения задач ранжирования, предсказания и построения рекомендаций использует разработанную ими систему машинного обучения CatBoost, основанную на градиентном бустинге. Бустинг (англ. boosting – улучшение) – это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов [4].

Технология CatBoost позволяет эффективно обучать модели на разнородных данных, при этом данные могут быть использованы в первоначальном виде, переводить их на язык цифр не требуется, что существенно влияет на точность работы модели.

Метод основан на двух популярных алгоритмах машинного обучения:

– дерево решений - это способ представления правил в иерархической, последовательной структуре, где каждому объекту соответствует единственный узел, дающий решение (под правилом понимается логическая конструкция, представленная в виде "если ... , то ...");

– нейронные сети (искусственная нейронная сеть) - математическая модель, её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей.

Выбор используемого алгоритма зависит от типа задачи и данных.

Персональные рекомендации на сервисе Яндекс.Маркет формируются с помощью системы CatBoost (рисунок 2). Точность рекомендаций говорит о способности алгоритма минимизировать проблемы масштабируемости, смещения результатов и снижения эффективности.

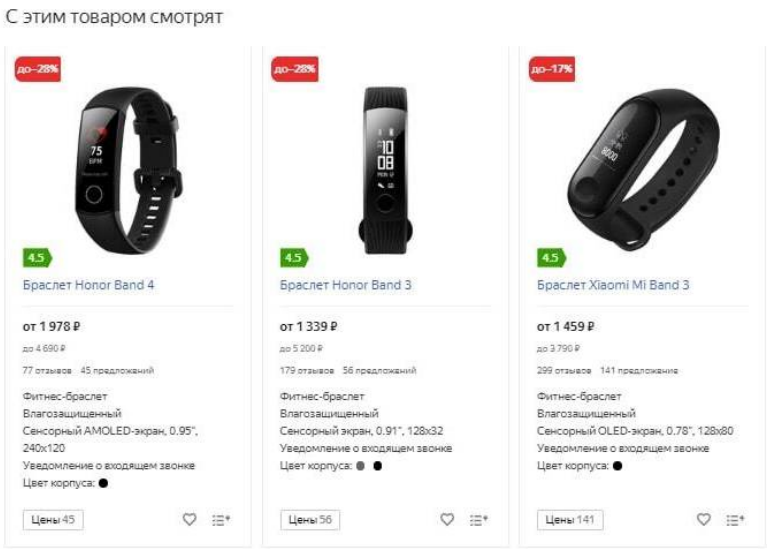


Рис. 2. Персональные рекомендации на Яндекс.Маркет

Технология CatBoost является запатентованной технологией Яндекс и используется, в первую очередь, самой компанией для улучшения результатов поисковой системы Яндекс, ранжирования ленты рекомендаций, расчета прогноза погоды и в других сервисах компании. Не смотря на открытый доступ к коду данного алгоритма, разработать на его основе рекомендательную систему для другой компании невозможно без получения на это особых прав.

Сервисы рекомендаций, рассмотренные выше, имеют ряд достоинств и широко применяются в рамках деятельности своих компаний. Алгоритмы, на основе которых были по-

строены данные сервисы, успешно решают ряд проблем, связанных с обработкой данных, но они принадлежат этим компаниям, и использовать их в полной мере для построения собственных сервисов рекомендаций другие компании не могут, кроме того рекомендательные системы часто привязаны к области интересов компаний-разработчиков. Это говорит о необходимости разработки крупными компаниями собственных алгоритмов формирования рекомендаций, которые будут способны предоставлять точные рекомендации в контексте удовлетворения потребностей конкретных предметных областей.

Библиографический список

1. Netflix Recommendations: Beyond the 5 stars (Part 1) [Электронный ресурс]. – URL: https://medium.com/netflix-techblog/netflix-recommendations-beyond-the-5-stars-part-1-55838468f429 (дата обращения: 11.02.2020).
2. Amazon opens up its product recommendation tech to all [Электронный ресурс]. – URL: http://www.engadget.com/2016/05/17/amazon-dsstne-ai-open-source/ (дата обращения: 11.02.2020).
3. Кежватова А. Т. Использование интеллектуального анализа данных в задаче формирования персональных рекомендаций клиентам ПАО Сбербанк. 66 студенческая научно-техническая конференция. Рязанский государственный радиотехнический университет имени В.Ф. Уткина. Материалы конференции. – 2019. – С. 48 – 49.
4. CatBoost – новый метод машинного обучения от Яндекса [Электронный ресурс]. – URL: https://yandex.ru/blog/company/catboost-novyy-metod-mashinnogo-obucheniya-ot-yandeksa (дата обращения: 11.02.2020).

УДК 528.9:911.52:57.2

ОЦЕНКА ПРИГОДНОСТИ ЗЕМЕЛЬНЫХ УЧАСТКОВ ДЛЯ РЕАЛИЗАЦИИ КОММЕРЧЕСКИХ ПРОЕКТОВ ПО ЦИФРОВОЙ КАРТЕ МЕСТНОСТИ

В.В. Булгаков[1], С.И. Елесина[1], Е.М. Никифорова[2]
[1]Рязанский государственный радиотехнический университет им. В. Ф. Уткина,
[2]Рязанский государственный университет им. С. А. Есенина,
Россия, Рязань

Аннотация. В статье рассматриваются геоинформационные системы и их применение для оценки пригодности земельных участков для реализации коммерческих проектов по цифровым картам местности.
Ключевые слова: цифровая карта местности, геоинформационные технологии, эстетическая привлекательность ландшафта.

EVALUATION OF LAND SUITABILITY FOR IMPLEMENTATION OF COMMERCIAL PROJECTS ON DIGITAL MAP OF THE AREA

Bulgakov V.V.[1], Elesina S.I.[1], Nikiforova E.M.[2]
[1]Ryazan state radio engineering university named after V.F. Utkin,
[2]Ryazan state university named after S. A. Yesenin,
Russia, Ryazan

Annotation. The article discusses geographic information systems and their application to assess the suitability of land for the implementation of commercial projects on digital maps of the area.
Keywords: digital map of the area, geographic information technology, aesthetic appeal of the landscape.

Введение

Во второй половине XX века дистанционные исследования планеты Земля достигли качественно нового уровня. Это стало возможным благодаря размещению специализирован-

ной аппаратуры на искусственных спутниках, пилотируемых космических кораблях и орбитальных станциях. К настоящему времени созданы технологии, которые позволяют получать космические снимки в широком диапазоне частот, различного масштаба и пространственного разрешения практически для любого участка поверхности нашей планеты. Информация, получаемая при обработке этих снимков, успешно используется при геоморфологических исследованиях земной поверхности с целью:

- составления региональных геоморфологических и тектонических карт;
- оптимизации поисковых работ на различные виды полезных ископаемых;
- проведения различных специализированных геологических, инженерно-геологических, гидрогеологических и экологических исследований;
- изучения геодинамических процессов различного генезиса.

Целью данной работы является исследование и разработка методики оценки и расчёта характеристик объектов на основе информации от цифровой карты местности.

Цифровые карты местности

Цифровая карта местности (или просто цифровая карта) является компьютерной моделью, которая содержит данные и правила, описывающие положение и пространственно-логические взаимоотношения объектов местности.

Цифровая карта – основа информационного обеспечения автоматизированных картографических систем (АКС) и географических информационных систем (ГИС), и является результатом их работы.

Цифровые карты непосредственно воспринимается человеком при визуализации электронных карт (например, на видеоэкранах), и компьютерных карт на твёрдом носителе. Они могут быть использованы как источник информации в машинных расчётах без визуализации в виде изображения.

Геоинформационные технологии (или ГИС-технологии) предназначены для создания цифровых моделей окружающего мира. Одно из основных направлений ГИС-технологий – это цифровое картирование. Основой для цифрового картирования являются аэрокосмические снимки, данные дистанционного зондирования, бумажные карты, которые с помощью ГИС-технологий могут быть переведены в цифровую форму. В результате обработки получается цифровая модель местности (ЦММ), представляющая собой матрицу, элементами которой являются либо высоты, либо трёхмерные координаты X, Y, Z. Математической основой преобразования является метод наименьших квадратов (НМК).

В начале XXI века специализированная обработка космических снимков позволила не только визуализировать линеаментную сеть планетарного масштаба, но и определить азимуты основных (главных) направлений этой сети – ортогонального (субмеридионального – 0-10° и субширотного – 80-90°) и двух диагональных – северо-восточного (30-60°, среднее 45°) и юго-восточного (130-140°, среднее 135°).

Цифровые карты местности (англ. DigitalTerrainMap, DTM) являются моделью карты местности, которая читается электронными средствами, и состоит из трёх параметров:

- географическая долгота (иногда меридиан, Latitude) – длина: 360° (от −180° до 180°) определяется начальной точкой в обсерватории Гринвича (Лондон). Соответствует оси *X* в геометрии;
- географическая широта (Latitude) – длина: 180°, в которой −90° соответствует Южному полюсу, а +90° – Северному. Соответствует оси *Y* в геометрии;
- географическая высота (Elevation) – в географии это высота над уровнем моря. Положительное значение соответствует горной местности, а отрицательное – глубине в водоёмах (чаще всего это океан). Соответствует оси *Z* в геометрии. В программных реализациях для работы с географическими объектами высота используется редко.

Классификация цифровых карт по содержанию и назначению эквивалентна общей классификации карт: цифровая топографическая карта, цифровая авиационная карта, цифровая геологическая карта, цифровая кадастровая карта, и другие.

Цифровые карты имеют главный недостаток – дороговизна. Так, для программы MosMap обновление карты Москвы с Московской областью стоит 20000 рублей, для Ленинградской области с центром в Санкт-Петербурге – 15000 рублей.

Для более информативной карты России с учётом населённых пунктов установка обходится в 40000 рублей, а для карт в открытом формате, таких как mapinfo, варьируется от 50000 до 300000 рублей, причём цена отличается от области, края или республики.

Методика измерения параметров рельефа земной поверхности по ЦКМ

Измерение площади озера. Измерить площадь озера можно с помощью различных ГИС, таких как GRASS, ScanEXImage, ERDAS и другие.

Один из простых способов измерения – это способ палетки, заключающийся в делении на несколько сегментов и расчёта площади озера по каждому сегменту, и используется в геометрических и математических расчётах.

Наиболее точный способ, но трудоёмкий – это составление по координатам. Составление по координатам заключается в создании многоугольника из произвольных координат (на которых существует граница между водным объектом и сушей), но последняя координата обязательно должна быть такой же, как у первой –получается замкнутый многоугольник (который может иметь от трёх точек). Как только составлен полученный многоугольник, можно определить площадь озера, моря или любого другого объекта.

Измерение глубины оврага. Измерить можно нивелированием. Нивелирование бывает разным: радиолокационное, геометрическое (при помощи нивелира и рейки), тригонометрическое, а также гидростатическое (на основе свойства жидкости сообщающихся сосудов и всегда находится на одном уровне вне зависимости от высоты точек, на которых были установлены сами сосуды, соединённые шлангами), барометрическое (при помощи барометра) и спутниковое (GNSS-приёмник).

Измерение расстояния между объектами. Большинство программ и сервисов на основе использования навигационных систем, включая Яндекс.Карты, GoogleMaps, MosMap, 2ГИС и ряд других, позволяют измерять расстояние через инструмент «Линейка». Как и измерение площади озера, существуют способы измерения, аналогичные озеру: составление по координатам, способ палетки и другие.

Оценка пригодности земельных участков для реализации коммерческих проектов

Оценка возможности строительства дороги до объекта. Для оценки возможности строительства существует 30 статья Земельного Кодекса РФ. До начала строительных работ нужно позаботиться о получении разрешительной документации на строительство в соответствии с градостроительным кодексом РФ. Согласно статье 51 Градостроительного кодекса РФ, необходимо согласовать то, что конкретно собирается построить застройщик. Разрешение на строительство – это документ, коим подтверждается, что проектная документация полностью соответствует требования, установленным градостроительным планом земельного участка. Имея на руках такой документ, застройщик получает полное право возводить дом (или другие объекты капитального строительства).

Для оценки возможности строительства дороги до любого объекта нужно определить географическое местоположение, вместимость и основные технологические характеристики, а также выбор площадки для строительства . Оценку такой способности можно по пригодно-

сти по ряду основных критериев:

- ***площадь участка***. Зависит от многих факторов и не имеет прямой либо пропорциональной привязки к объёму объекта, так как можно желаемый объём набрать этажностью;
- ***целевое назначение земель***. Желательно, чтобы назначение было «земли промышленности». В противном случае изменение целевого назначения довольно хлопотное дело;
- ***геология участка***. Если участок находится в собственности застройщика, то до начала проектирования в любом случае придётся производить инженерно-геологические изыскания. Если рассматривается вопрос о приобретении участка, то желательно иметь результаты таких изысканий, так как неблагоприятные геологические условия могут повлечь за собой значительное удорожание проекта. В первую очередь это отразится на конструктивном исполнении фундаментов и, как следствие, на объёмах и стоимости фундаментальных работ;
- ***геодезические изыскания***. Ситуация, подобная геологии. Визуально незначительный перепад высотных отметок в результате изысканий может составлять несколько метров, и позже вылиться в многотысячные объёмы земляных работ. Стоимость этих работ может быть настолько большой, что могут появиться мысли о поиске новой площадки;
- ***подъездной железнодорожный путь***. Если предполагается отгрузка зерна на железнодорожный транспорт, то здесь обстановку с возможностями подъездных путей лучше выяснить с работниками железной дороги, например, с начальником станции, прилегающей к участку. Его просто нужно ознакомить с намерениями застройщика, и он подскажет, что и как лучше сделать;
- ***автомобильные подъездные дороги***. Поскольку многофункциональные здания предусматривает значительный поток автомобильного транспорта, наличие подъездных дорог с твёрдым покрытием просто необходимо. И если многофункциональное здание будет строиться в поле в 5 километрах от существующих автомобильных дорог, нужно не забывать включить в стоимость объекта ещё и стоимость строительства пяти километров дороги;
- ***газификация***. Тоже придётся обратиться к соответствующим специалистам в местную газовую службу. Результатом обращения является положительный ответ от газовой службы своего района на возможность подключения к её сетям и понимание удалённости этого подключения к участку;
- ***электрификация***. Обеспечение качественным гарантированным и бесперебойным электропитанием является основной задачей комплексного обеспечения для многофункционального здания;
- ***водоснабжение***. На многофункциональном здании потребуется вода для хозяйственно-питьевых нужд рабочих и служащих, а также для целей пожаротушения. Чаще всего это будет центральный водопровод, а с другой стороны придётся бурить водозаборную скважину;
- ***наличие на участке зданий и сооружений***. Если на участке существуют другие строения, которые могут быть использованы для обустройства административно-бытового корпуса, весовой, лаборатории или мастерских, этот участок является наиболее привлекательным;
- ***эстетическая привлекательность ландшафта***. Этот критерий наиболее актуален при планировании строительства отеля, базы отдыха, санатория, развлекательного центра и т.п. В России так много красивых мест, что строить сооружения подобного назначения без учета этого фактора не разумно, а, следовательно, и проектировать трассу. Эстетическая привлекательность территории так же может быть оценена с применением геоинформационных методов. Наиболее часто используют сервисы ArcGIS и QGIS Desktop. Они по-

зволяют получить количественные оценки, позволяющие более эффективно производить сравнительный анализ и выбор территории. С помощью свободной географической информационной системы с открытым кодом QGIS Desktop можно не только создавать и редактировать, но и осуществлять визуализацию и анализ геопространственной информации. Построенные таким образом на основе данных цифровой карты местности синтезированные обзорные панорамы и зоны видимости с различных точек наблюдения позволяют проектировщику получить визуальное представление о ландшафте и произвести его субъективную оценку.

Таким образом, используя современные геоинформационные системы, взвесив перечисленные выше основные характеристики участка, оценив все «за» и «против», Застройщик может принять решение о его пригодности либо непригодности к строительству. Если «за» преобладает, то следующим этапом является предварительная автоматизированная прокладка трассы дороги до предполагаемого объекта. Наибольшую трудность вызывает трассировка в сложных географических условиях: в зоне холмов, гор, оврагов, ущелий и т.п.

Методика трассирования дорог в зоне оврагов. Выбор направления трассы дорог в овражистой местности в значительной степени определяется положением населенных пунктов, между которыми прокладывается дорога, и планом овражной сети.

Рациональное направление трассы выбирают в зависимости от конфигурации овражной сети и категории дороги. При развитой сети оврагов положение дороги в обход оврагов снижает стоимость постройки, но приводит к большой извилистости трассы, перепробегу автомобилей. Поэтому дороги высших категорий следует прокладывать, приближаясь к кратчайшему направлению, не допуская, конечно, излишних пересечений оврагов. Увеличение объемов земляных работ и числа искусственных сооружений оправдывается в этом случае значительным удешевлением стоимости проезжей части и сокращением дорожно-транспортных расходов.

При проектировании дорог низших категорий, особенно сельских, решающим фактором является стоимость строительства. В этом случае более выгодным оказывается вариант трассы, запроектированный с минимальным числом пересечений оврагов.

При обходе оврага трассу располагают на расстоянии 50-100 м от его вершины, обязательно предусматривая в проекте мероприятия по закреплению оврага. Располагать трассу непосредственно вблизи вершины, в зоне размыва нерационально, так как потребуются дополнительные работы по защите земляного полотна и сооружения от размыва.

При направлении трассы вдоль речной долины не следует прокладывать дорогу по конусам выноса пересекаемых оврагов, где обычно наблюдается блуждание русла водотока. При большой интенсивности отложения наносов отверстие искусственного сооружения может быть занесено. Наиболее целесообразно пересекать овраги выше конуса выноса в пределах транзитной зоны оврага. Однако если мелких оврагов очень много, трассу, чтобы не удлинять, прокладывают по конусам выноса, предусматривая для упорядочения протекания воды устройство подходных русел и дамб, направляющих поток в отверстие моста и предохраняющих земляное полотно от размыва, а отверстие искусственного сооружения от засорения наносами. При пересечении широких и глубоких балок иногда приходится развивать трассу по их склонам для уменьшения объемов земляных работ. Автомобильные магистрали с интенсивным движением пересекают глубокие долины и овраги большими виадуками в уровне их краев. Длина дороги сокращается, а автомобилям не приходится спускаться к расположенному на дне долины низкому мосту, а затем вновь подниматься вверх по склонам.

В овражистых районах, на участках дорог с большими продольными уклонами особое внимание следует обращать на укрепление боковых и водоотводных канав. В пылеватых и суглинистых грунтах обычная канава при размыве может быстро превратиться в овраг, разрушающий дорогу.

Заключение

Рассмотренная в статье методика измерения параметров рельефа земной поверхности по ЦКМ позволяет производить оценку пригодности земельных участков для реализации коммерческих проектов.

Библиографический список

1. Градостроительный кодекс РФ № 190–ФЗ от 29.12.2004
2. Земельный кодекс Российской Федерации № 136–ФЗ от 25.10.2001
3. СП 11-102-97. Инженерно-экологические изыскания для строительства. Режим доступа: http://docs.cntd.ru/document/871001220
4. Krivtsov, Vyacheslav A. The information technology of the terrain attractiveness assessment / Vyacheslav A. Krivtsov, Elena M. Nikiforova, Natalya V. Akinina, Pavel V. Belyakov // Mediterranean conference on embedded computing (MECO-2019), – Montenegro, Budva, june 10 – 14, 2019. – PP. 791-794
5. Никифорова Е.М. Эстетические свойства рельефа на ряде участков Рязанской области и их влияние на рекреационный потенциал территории.//Тенденции и проблемы развития индустрии туризма и гостеприимства: материалы 5-й Межрегиональной научно-практической конференции с международным участием,15 ноября 2018г./Ряз.гос.ун-т им. С.А.Есенина – Рязань, 2018 г. – С. 29 – 33.
6. Коберниченко, В.Г. Радиоэлектронные системы дистанционного зондирования Земли [Электронный ресурс] : учебное пособие / В.Г. Коберниченко. — Электрон. дан. — Екатеринбург :УрФУ, 2016. — 220 с. — Режим доступа: https://e.lanbook.com/book/99060.
7. Бибаева А.Ю., Макаров А.А. Применение ГИС для расчета комплексных показателей эстетической оценки ландшафтов//Известия Иркутского государственного университета. Серия Науки о Земле. 2018. Т.24. С. 17-33.https://doi.org/10.26516/2073-3402.2018.24.17
8. Ротанова И.Н., Васильева О.А. Оценка эстетической привлекательности ландшафтов проектируемого природного парка «Предгорье Алтая» с применением геоинформационных технологий//Наука и туризм: стратегии взаимодействия. 2017 (5). С. 29-36

УДК 004.9; ГРНТИ 50.41

АВТОМАТИЗАЦИЯ ПРОЦЕССА ПОЛУЧЕНИЯ ГОТОВОЙ ПРОДУКЦИИ ПОТРЕБИТЕЛЯМИ

А.А. Рунцо

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, nastya.runtso@yandex.ru

Аннотация. В работе рассматриваются проблемы предприятия, которые связаны с реализацией готовой продукции конечным потребителям и информационные системы в данной области. Приводятся основные особенности ИС, достоинства и недостатки.

Ключевые слова: конкурентоспособность, автоматизированная система управления складом (АСУС), система управления складом «Vector», система SolvoWMS, система CoreWMS.

AUTOMATION OF THE PROCESS OF RECEIVING FINISHED PRODUCTS BY CONSUMERS

A.A. Runtso

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, nastya.runtso@yandex.ru

The summary. The paper discusses the problems of the enterprise that are associated with the sale of finished products to end consumers and information systems in this area. Given main features of IP, advantages and disadvantages.

Keywords: competitiveness, computerized warehouse control system (ASUS), warehouse control system «Vector», SolvoWMS system, CoreWMS system.

На предприятиях существуют проблемы, которые негативно сказываются на взаимоотношениях с потенциальными потребителями продукции и снижают его конкурентоспособность [1,5]. Поэтому особое внимание должно уделяется не только производственному процессу, но и обслуживанию потребителей. Среди таких проблем можно выделить следующие:

- заказ продукции осуществляется через контактный телефон предприятия;
- неготовность продукции к выдачи потребителям;
- неинформированность потребителей о точной дате готовности продукции;
- медленный процесс загрузки готовой продукции;
- долгое ожидание продукции потребителями, что приводит к скоплению машин на территории предприятия.

Для устранения простоев в обслуживании предполагается внедрение автоматизированной информационной системы для повышения эффективности процесса получения готовой продукции потребителями. Потребителям будет удобно производить заказы продукции у предприятия. Требования к информационной системе:

1) возможность оформить заявку на получение готовой продукции предприятия;
2) получение сведений о готовности своего заказа;
3) СМС рассылка о возможности приезда на предприятие за готовой продукцией.

Рационально сформированная система оповещения необходима для того, чтобы предприятию удержать постоянных клиентов продукции, повысить привлекательность среди новых потенциальных потребителей, создать благоприятные отношения к предприятию. Установление крепких долгосрочных взаимоотношений с потребителем позволит предприятию повысить конкурентоспособность, так как ориентация на клиента является ключевым инструментом по достижению конкурентных преимуществ.

Реализация продукции должна осуществляется по четкой отлаженной системе, начиная с процесса оформления заявки на отгрузку готовой продукции до полного завершения загрузки продукции клиенту. Исходя из этого, предприятию необходимо своевременно информировать клиентов о готовности продукции или ее отсутствии на данный момент времени, тем самым позволяя не допускать конфликтных ситуации и способствовать улучшению качества обслуживания.

Таким образом, разрабатываемая информационная система будет представлять собой приложение, в котором каждый потребитель сможет оформить заявку на получение готовой продукции, узнать о готовности своего заказа и получать СМС рассылку о возможном приезде на предприятия за получением готовой продукции. Каждый товар в каталоге официального сайта предприятия описывается типом продукции, техническими характеристиками, логистическими характеристиками, способом применения, сертификатами и документацией. Покупатель может загрузить прайс-лист каталога продукции (документ, в котором описывается продукция, цена и метраж).

Ранее заказ происходил через контактный телефон предприятия, теперь потенциальные потребители смогут воспользоваться приложением. После оформления заказа данные направляются в отдел продаж, где они обрабатываются и направляются в производственно-технический отдел для производства необходимой продукции. После того как продукция будет готова и доставлена в транспортно-складской комплекс, сотрудник отдела продаж сообщает покупателю о готовности продукции и времени приезда на предприятие. Далее покупатель сможет на своем транспорте приехать и забрать готовую продукцию со склада предприятия.

Далее рассмотрим информационные системы, которые связанны с реализацией готовой продукции конечным потребителям.

На сегодняшний день существует большое количество автоматизированных информационных систем, связанных с процессом реализации готовой продукции: ERP-система, CRM-система, SCM-система, Класс365, система управления складом «Vector», автоматизированная система управления складом (АСУС), CoreWMS, SolvoWMS [3]. Данные информационные системы обладают не полным функционалом, который необходим для повышения эффективности процесса получения готовой продукции (требования: возможность оформить заявку на получение готовой продукции предприятия, получение сведений о готовности своего заказа, СМС рассылка о возможности приезда на предприятие за готовой продукцией), который необходим для повышения эффективности процесса получения готовой продукции. Рассмотрим некоторые из них.

Автоматизированная система управления складом (АСУС). Представляет собой систему, осуществляющую процессы планирования и контроля приемки, хранения и контроля готовой продукции со склада. Система управления складом состоит из разнообразных функциональных подситем, которые управляют запасами, технологическими процессами, контролируют поставки [6]. При работе с АСУС осуществляется бухгалтерский учет, а именно расчеты с поставщиками и потребителями продукции, поступление, учет и выдача готового заказа со склада предприятия, оперативное управление установления хозяйственных связей между предприятием и потребителями продукции и комплектаций и управленея складскими механизмами.

Система управления складом «Vector». При внедрении данной системы на предприятия происходит охват всех аспектов управления предприятием, а именно управление складским механизмом с четко отлаженной работой.

Автоматизация складских процессов при ситеме управления «Vector» осуществляется в таких операциях как:

1) приём продукции;
2) контроль качества входной продукции;
3) размещение товара;
4) осуществление комплектации заказов готовой продукции;
5) контроль качества выходного заказа продукции;
6) работа с документами сертификата качества;
7) отгрузка готовой продукции;
8) перескладирование и перераспределение;
9) инвентаризация;
10) формирование отчётной документации [4].

Преимущества: контроль движения каждой единицы товара, точность исполнения заданной процедуры, контроль качества выполнения, модульная архитектура.

Недостатки: неоповещение о готовности товара.

SolvoWMS – одна из лидирующих WMS-систем, которая активно развивается более 20 лет. Относится к классу конфигурируемых систем управления для складов любого типа и назначения [6].

Преимущества: производительность системы, автоматизированный процесс приемки, автоматизированное размещение груза, технология «Удаленный склад», искусственный интеллект, гибкая система отчетности.

Недостатки: обладает не полным функционалом для ЗАО «МПК «КРЗ».

CoreWMS. Система CoreWMS разработана с учетом современных технологий управления складами, имеет модульную структуру, быстро настраивается, надежно работает и предоставляет возможность эффективно использовать самые передовые информационные технологии. В основном модуле системы (ядре) реализована информационная поддержка

всех основных складских операций: приема, размещения и перемещения товаров по складу, инвентаризации, отбора и отгрузки товаров [2]. Функциональность системы легко расширяется и настраивается по желанию заказчика за счет дополнительных модулей.

Преимущества: активно развивается с учетом потребностей пользователей, надежная, высокопроизводительная система, способность обеспечить управление любым складом.

Недостатки: аналогичны недостаткам SolvoWMS.

Все рассмотренные ИС обладают схожей функциональностью. Их применение может обеспечить необходимый уровень удовлетворения требований, предъявляемых ИС предприятием-заказчиком, это повысит конкурентоспособность по фактору обслуживание.

Исходя из этого, произойдет увеличение потенциальных заказов на производимую предприятием продукцию. Потенциальные заказчики перестанут уходить к конкурентным фирмам за более комфортные условия заказа и выдачи готовой продукции. Рационально сформированная система оповещения позволит предприятию удержать постоянных клиентов продукции, повысить привлекательность среди новых потенциальных потребителей, создать благоприятные отношения к предприятию. Установление крепких долгосрочных взаимоотношений с потребителем позволит предприятию повысить конкурентоспособность, так как ориентация на клиента является ключевым инструментом по достижению конкурентных преимуществ.

Библиографический список

1. Петрушенкова Т.М. ЗАО «МПК «КРЗ». 55 лет вместе с отраслью и страной. – М.: Издательство «Юнион Принт», 2017. – 16 с.

2. Автоматизированная система управления складом [Электронный ресурс]. – Режим доступа: https://mirznanii.com/a/164035-3/avtomatizirovannaya-sistema-upravleniya-skladom-3/.Дата обращения 11.02.2020 г.

3. Анализ существующих ИТ-решений в области реализации готовой продукции [Электронный ресурс]. – Режим доступа: https://studbooks.net/2270134/informatika/analiz_suschestvuyuschih_resheniy_oblasti_realizatsii_gotovoy_produktsii. Дата обращения 09.01.2020 г.

4. Системы «Vector», (АСУС), CoreWMS [Электронный ресурс]. – Режим доступа: https://studwood.ru/898201/marketing/sistemy_vector_asus_corewms. Дата обращения 11.02.2020 г.

5. Официальный сайт ЗАО «МПК «КРЗ» [Электронный ресурс]. – Режим доступа: http://www.krz.ru. Дата обращения: 03.02.2020 г.

6. SolvoWMS – система управления складом [Электронный ресурс]. – Режим доступа: https://www.solvo.ru/products/solvo-wms/?_openstat=ZGlyZWN0LnlhbmRleC5ydTsxNzIzOTM4OzQ4MTQxOTk7d3d3LnlhbmRleC5ydTpwcmVtaXVt&yclid=7372409214677049894. Дата обращения 11.02.2020 г.

УДК 004.9; ГРНТИ 50.01

АВТОМАТИЗАЦИЯ ПРОЦЕССА УЧЁТА МАТЕРИАЛЬНЫХ ЦЕННОСТЕЙ ОРГАНИЗАЦИИ

Д.Н. Лошкарева

Рязанский государственный радиотехнический университет имени В.Ф,Уткина, Российская Федерация, Рязань, dasha.loschkareva@yandex.ru

Аннотация. В статье рассматриваются возможность автоматизации процесса учета товарно-материальных ценностей, которая позволит повысить качество и оперативность обработки документов учета, а также систематизировать информацию об объектах для полного отражения достоверной информации. Рассмотрено функциональное моделирование процесса учета ТМЦ.

Ключевые слова: автоматизация, товарно-материальные ценности, функциональное моделирование, хранилище, базы данных.

AUTOMATION OF THE PROCESS OF ACCOUNTING FOR MATERIAL ASSETS OF THE ORGANIZATION

D.N. Loshkareva
Ryazan State Radio Engineering University named after V. F. Utkin,
Russian, Ryazan, dasha.loschkareva@yandex.ru

The summary. The article discusses the possibility of automating the process of accounting for inventory items, which will improve the quality and efficiency of processing accounting documents, as well as systematize information about objects to fully reflect reliable information. Functional modeling of the process of accounting for goods and services is considered.
Keywords: automation, inventory, functional modeling, storage, databases.

Оперативная обработка значительных потоков информации является неотъемлемой задачей любой современной финансовой организации. Автоматизация процессов учета товарно-материальных ценностей позволяет сокращать время на обработку и оформление документов, проведение операций, выполнения расчетов и дальнейшее сопровождение объектов учета. [1].

Для создания информационной системы учета товарно-материальных ценностей в основу берутся этапы, представленные на рисунке 1.

Рис. 1. Этапы создания информационной системы учета ТМЦ

В исследуемой организации в учете товарно-материальных ценностей в большей степени задействованы два структурных подразделения. Это административно-хозяйственный отдел (в роли кладовщика) и отдел учета внутренних операций (в роли бухгалтера). Первоначальное распределение служебных обязанностей выглядит примерно следующим образом. От контрагента поступают товарно-материальные ценности, представленные основными средствами, нематериальными активами и инвентарем. Кладовщик проверяет первичную документацию на наличие или отсутствие фактических отклонений, правильность и достоверность предоставленных документов. В случае соответствия всех документов принимает объекты на склад, а далее коммуницирует с бухгалтером для постановки вторым на учет поступивших ценностей. Далее, бухгалтер занимается оформлением инвентарных карточек основных средств и нематериальных активов, ведет расчет амортизации объектов, составляет в архив ведомости по расчету амортизации; формирует журнал, полученных счет-фактур, а также отправляет статистическую отчетность по установленным формам.

Еще одной функциональной обязанностью кладовщика является выписка требований на товарно-материальные ценности со склада. На данный момент этот процесс не автоматизирован. В банке ведется книга учета, в которой кладовщик вручную вносит изменения и производит выдачу со склада. Этот процесс является трудозатратным, поскольку периодически нужно проводить инвентаризацию, сверку остатков и выполнять регулярные однотипные действия, которые можно упростить с помощью автоматизации процесса учета и уменьшить трудозатраты на их выполнение.

В организации, в связи с определенными обстоятельствами, планируется сокращение сотрудников административно-хозяйственного отдела и переноса обязанностей по ведению склада и учета товарно-материальных ценностей на бухгалтерию, при том же составе персонала в данном отделе. Технически, в случае добавления функционала бухгалтеру, объем выполняемой работы становится более трудоемким. Поскольку процесс учета товарно-материальных ценностей является сложным, необходимо автоматизировать процессы обработки и сопровождения поступивших документов и работу склада.

Стоит отметить, что для организаций, занимающихся банковской деятельностью, основной задачей является привлечение активов и наращения кредитного портфеля, в связи с чем затраты, связанные с приобретением программного обеспечения для учета товарно-материальных ценностей не являются в конечном итоге факторами, увеличивающими прибыль организации. Поэтому разработка своей информационной системы по работе с первичными документами учета ценностей позволит разгрузить функционал сотрудника по учету и является актуальной для предприятия.

Рассмотрим с помощью функционального моделирования модели IDEF0, учет товарно-материальных ценностей (ТМЦ) как совокупность взаимодействующих работ или функций (рис. 1).

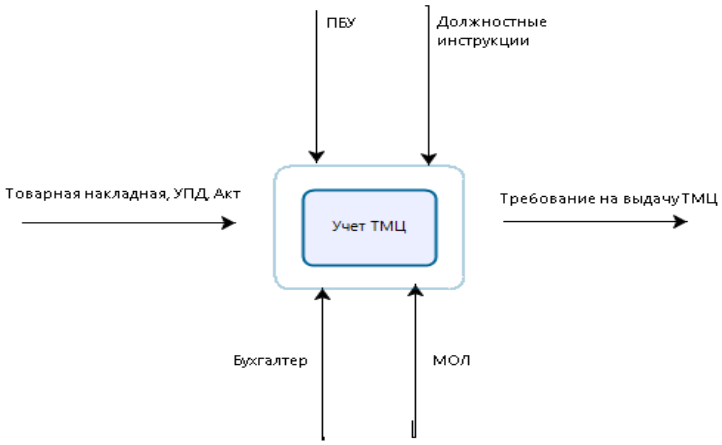


Рис. 2. Функциональная модель учета материальных ценностей

Процесс учета ТМЦ включает в себя такие основные операции по движению, как: поступление, перемещение, списание и другие. Все функции учета выполняется с минимальным использованием средств автоматизации. Процесс учета материальных ценностей осуществляется на основе первичных документов. Товар поступает на склад банка на основании товарной накладной, универсально-передаточного документа или акта на право использования. Далее, происходит процесс формирования требования на отпуск (выдачу) ТМЦ по правилам ПБУ, вписывается бухгалтером для материально ответственного лица и формирует документ на печать. Так, исходными данными являются первичные документы (стрелка входа); результатом является акт-требование на ТМЦ (стрелка выхода); исполнителями в данном случае будут бухгалтер и материальное ответственное лицо (стрелки механизма); средства исполнения – это должностные инструкция и сами положения бухгалтерского учета (стрелки управления).

Проделав ряд операций в автоматизированной системе, бухгалтер получит автоматически сформированный документ акт-требование, в котором будет указана вся необходимая информация о товарно-материальных ценностях. Система будет хранить данные как по основным средствам, так и по нематериальным активам. Данные о движении будут фиксироваться в программе, что позволит сократить количество возможных ошибок обработки пер-

вичных документов. Для своевременной работы необходимо структурировать данные по совершаемым операциям. Это позволяет планировать текущие действия и дает возможность замещения сотрудников. Так как в случае передачи дел или введение в курс работы нового служащего, система отобразит все необходимые данные, что в свою очередь также позволит взаимозаменить сотрудников на время их отсутствия.

Для автоматизации процесса учета товарно-материальных ценностей будут использоваться полученные профессиональные умения и навыки в сфере программно-информационных систем. Так, в системе 1С: Предприятие, включающей в себя платформу и прикладные решения, разработанные на ее основе для деятельности организаций, будут реализованы справочники, для хранения информации и создание форм отчетных документов. А так же для начала работы с данными будет использоваться MS Access, с помощью которых будут храниться все таблицы базы данных и для пользователя будет возможно создавать необходимые запросы и отчёты [2].

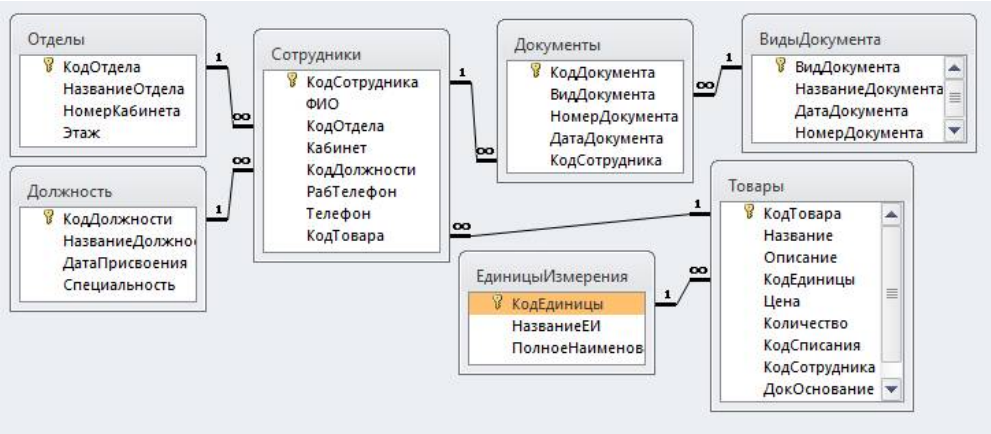


Рис. 3. Фрагмент схемы данных для процесса учета товарно-материальных ценностей.

В процессе создания информационной системы для автоматизации процесса учета товарно-материальных ценностей будут решены основные задачи, а именно обработка первичной документации в программе, автоматизированная функция выписки требований на соответствующие объекты, создание отчетов по движению и остаткам основных средств и нематериальных активов. В дальнейшем, можно усовершенствовать сформированную систему, для отражения наибольшего объема полезной информации, разработать удобный интерфейс и печатные формы отчетных документов. Полученная система позволит снизить временные затраты бухгалтера на обработку документов, снизить риски по возможным ошибкам учета. База данных будет являться доступной для понимания конечного пользователя с возможностью допрограммирования информационной системы автоматизации процесса обработки и сопровождения документов учета материальных ценностей.

Библиографический список

1. Вдовин В.М. Информационные технологии в финансово-банковской сфере: Учебное пособие / В.М. Вдовин, Л.Е. Суркова. - М.: Дашков и К, 2013. - 304 с.

2. Гончаров Д.И., Хрусталева Е.Ю. Решение специальных прикладных задач в 1С:Предприятии 8.3.: учебник для студентов бакалавриата и магистратуры: Пресс– сервис, 2016.

УДК 004.41; ГРНТИ 50.05.13

ПОВЫШЕНИЕ УРОВНЯ АБСТРАКЦИИ РАЗРАБОТКИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ С ИСПОЛЬЗОВАНИЕМ ВИЗУАЛЬНОГО МЕТОДА

А.И. Николаев

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, andrey.nikolaev130@gmail.com

Аннотация. В работе рассматриваются возможности повышения уровня абстракции, с которой работает программист при разработке параллельных программ. Рассмотрены примеры реализации, которые встречаются при использовании потоков в высокоуровневых языках программирования. Рассматривается возможность использования визуального метода с целью повышения уровня абстракции.

Ключевые слова: параллельное программирование, уровень абстракции, визуальный метод, поток, визуализация, мгогопоточность.

INCREASING THE LEVEL OF ABSTRACTION OF THE DEVELOPMENT OF PARALLEL PROGRAMS USING THE VISUAL METHOD

A.I. Nikolaev

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, andrey.nikolaev130@gmail.com

The summary. The paper considers the possibilities of increasing the level of abstraction with which the programmer works when developing parallel programs. Implementation examples that are encountered when using streams in high-level programming languages are considered. The possibility of using the visual method to increase the level of abstraction is considered.

Keywords: parallel programming, level of abstraction, visual method, thread, visualization, multithreading.

Использование параллельных программ ускоряет работу различных алгоритмов путем распараллеливания вычислений. Написание параллельных программ довольно сложная задача, особенно для разработчиков, которые только начитают свое знакомство с разработкой параллельных алгоритмов [1]. При написании параллельных программ требуется хорошее владение механизмами языка, в противном случае возрастает риск ошибки в алгоритме, и найти и устранить неточность, в случае многопоточных вычислений достаточно тяжело, а иногда невозможно без использования дополнительных средств [2]. Так же возникает проблема в скорости работы, если хоть один поток будет работать крайне медленно, то все остальные будут его ожидать и это приведет к тому, что наша программа будет похожа на последовательное выполнение команд, как в случае с однопоточной системой [3].

Для упрощения написания параллельных программ в высокоуровневых языках вводят различные уровни абстракции, так как детали реализации могут быть крайне сложные для восприятия человеком. Так на самом нижнем уровне машина оперирует нулями и единицами. Так как это не совсем удобно и легко воспринимаемо человеком, был придуман язык ассемблера, что является символьным представлением двоичного языка компьютера. С течением времени появились более высокоуровневые языки программирования, такие как Java, C# и т.п. В этих языках программист пишет текст программ с использованием операторов языка. Разработчику уже нет необходимости править двоичные коды программ, все это возлагается на компиляторы и трансляторы, которые переводят текст программ на язык понятный машине. Если рассмотреть подробнее устройство Java, то в ней присутствует виртуальная машина, которая реализована для различных операционных систем, это позволяет запускать одни и те же программы в абсолютно разных средах. За счет всего этого повышается уровень абстракции, с которой работает программист.

Абстракция в системах обычно делится на несколько уровней и на каждом уровне абстракция должна быть согласована. Таким образом вся система разделяется на несколько уровней, которые могут работать с другими уровнями относительно автономно. И так же

программисту нет необходимости держать в голове информацию об абстракциях других уровней. Абстракции помогают в борьбе со сложностью систем. Они так же уменьшают связанность разрабатываемых программ и обеспечивают взаимозаменяемость различных компонентов.

При разработке параллельных программ с использованием высокоуровневого языка, программист не работает напрямую с потоками в системе. Так, например, в языке программирования Java есть свои классы и методы для работы с потоками, программисту нет необходимости напрямую взаимодействовать с потоком, это оставлено для уже существующих в языке средств. Пример подобных абстракций – класс Thread, у которого есть набор необходимых методов для работы с потоком. Программист может работать с высокоуровневой абстракцией, не опускаясь в детали, это позволяет снизить сложность разрабатываемых алгоритмов, а также уменьшить количества ошибок допущенных при разработке программных систем с использованием параллелизма. При необходимости опытный разработчик спуститься на несколько уровней абстракции ниже, и сделать более точные настройки, но это необходимо в исключительных ситуациях. При обычном использовании механизмов много поточности для построения параллельных программ достаточно использования высокоуровневой абстракции, которую предоставляет язык программирования.

Использование возможностью управления потоков для написания параллельных программ на низком уровне увеличивает время необходимую на разработку и отладку новых алгоритмов, при этом ухудшается качество, так как велика вероятность ошибки. Так же очень тяжело разбираться новому программисту в уже написанном коде и это увеличивает сложность поддержки и обслуживания существующего кода. Помимо увеличения времени и сил на разработку, так же повышается уровень входа для начинающих разработчиков, что так же негативно влияет на данное направление. Для написания эффективных параллельных алгоритмов требуются более опытные разработчики, которые стоят также дороже и их количество обычно ограничено. Так же увеличиваются затраты на подготовку новых специалистов. Для обучения требуется больше времени и так же требования к квалификации преподавателей тоже растет. Гораздо эффективнее работать с высокоуровневыми абстракциями и не нарушать работу на низком уровне.

Рассмотрим и сравним существующие механизмы, которые существуют в языка Java и C# для написания параллельных программ и работы с потоками. В языке программирования Java главным для работы с потоками является класс Thread [4]. Он имеет в себе следующий методы:

- getName() - получить наименование потока;
- getPriority() - получить приоритет потока;
- isAlive() - определить, находится ли поток в состоянии выполнения;
- join() – приостановить выполнение текущего потока и ожидать окончание выполнения потока;
- run() - внутри этого метода пишется текст программы, который должен выполняться в отдельном потоке;
- sleep() – поставить поток на паузу на указанное время;
- start() - запустить выполнение потока.

Разработчик оперирует данными методами и не спускается на абстракцию ниже для того, чтобы посмотреть, как это работает, ему нет необходимости это знать. Но для большинства методов синхронизации, таких как семафор, флаг, мьютекс, критическая секция, событие, замок с обратным отсчетом, обменник достаточно стандартных механизмов языка [5].

В языке программирования C# так же существует класс Thread и у него есть методы и свойства, которые являются эквивалентами методов в Java:

- Name – получить имя потока, который является его идентификатором;

- Priority – получить приоритет потока;
- IsAlive() - определить, находится ли поток в состоянии выполнения;
- Join() - приостановить выполнение текущего потока и ожидать окончание выполнения потока;
- Sleep() – поставить поток на паузу на указанное время;
- Start() - запустить выполнение потока.

Отличием является метод run(). В случае с Java текст программы, который необходимо выполнить, пишется в этом методе. В C# текст программы, который необходимо выполнить, передается в конструкторе при создании. Механизмы синхронизации можно описывать как и в Java с помощью стандартных операторов языка.

В большинстве высокоуровневых языков существуют абстракции для работы с потоками, это позволяет использовать визуальный метод создания параллельных программ совместно с кодогенерацией. Мы можем один раз нарисовать необходимое нам поведение и сформировать код на конкретном языке программирования за счет модулей преобразования внутреннего представления визуального отображения. Так как во многих языках существуют похожие конструкции для работы с потоками этот процесс формирования кода будет возможен для различных языков. В случае сильного отличия придется просто доработать соответствующий модуль преобразования. За счет того, что у визуального представления имеется единое внутреннее интерпретация с определенной структурой, эту структуру можно проверять на соответствие правилам формирования параллельных программ, что позволит избежать ошибок еще до этапа генерации текста программы. Необходимо установить четкие правила по которым внутреннее представление будет преобразовываться в ту или иную конструкцию языка и эти правила должны быть эквивалентными для всех языков, для которых поддерживается кодогенерация.

Использование разработчиком кодогенерации из внутреннего представления позволяет оперировать различными представлениями предметной области или задачи. Так как внутреннее представление помимо текста программы может быть отображено в каком-либо удобочитаемом и удобно редактируемом формате. Одним из таких представлений является визуальное представление. В этом случае параллельный алгоритм строится в визуальную картинку, с которой работает программист. Это позволяет повысить уровень абстракции и сосредоточится на решении поставленной задачи и не отвлекаться на детали реализации. Так же этот подход может иметь положительное воздействие при обучении. Человеку, которые впервые сталкивается с понятиями поток и параллельные алгоритмы достаточно тяжело это понять с первого раза. Гораздо проще воспринимать подобную информацию с использованием визуального представления. Это позволит дать студентам в вузе базовое представление о работе параллельных программ за более короткий срок. При разработке крупных систем, визуализацию можно использовать при необходимости разработки каких либо параллельных алгоритмов с высокой долей абстракции. При этом будет сгенерирован текст программы в визе каркаса, которые останется наполнить бизнес логикой. Все необходимые синхронизации между потоками уже будут настроены. Это позволит сильно сократить время разработки и повысить качество. Кроме того это позволит представить сложные алгоритмы в более наглядном виде, что может способствовать повышению качества передачи знаний. При разработке крупных систем разработчики и архитекторы могут меняться и необходимо передавать все полученные знания своим преемникам, для этого служит документация. Визуальный метод является отчасти самодокументирумым, так как представляется в наглядном виде.

Кроме рассмотренных преимуществ в использовании абстракции высокого уровня есть некоторые недостатки. При необходимости внести изменение на каком либо из уровней абстракции, эти изменения приходится вносить во все остальные промежуточные уровни. Так же использование абстракции от источников данных может дать неоптимальный вариант использования этого источника, не учитывается специфика работы с необходимым источни-

ком, кроме того иногда достаточно неочевидно, как будет строится текст программы по визуальному представлению и достаточно тяжело с этим разбираться. Частично эти проблемы возможно решить добавлением большего количества метаинформации для модели при визуальном проектировании параллельных программ.

Данный подход рассматривается как продолжение исследований развивающихся методов программирования [6-7].

Библиографический список

1. Богачев К .Ю., Основы параллельного программирования. – М.: БИНОМ. Лаборатория знаний, 2014 – 342 с.

2. McConnell S. C. Code complete, 2nd Edition. – Redmond, Washington: Microsoft Press, 2004 – 914 p.

3. Николаев А.И. Разработка визуального метода проектирования параллельных программ // Новые информационные технологии в научных исследованиях – НИТ-2018: материалы XXII Всероссийск. научн.-техн. конф. студентов, молодых ученых и специалистов./Рязан.гос.радиотехн.ун-т. – Рязань, 2018. – С. 182-183.

4. Oaks S., Wond H. Java Threads, Third Edition. – Sebastopol: O'Reilly Media, 2009 – 360 p.

5. Николаев А.И. Визуализация параллельного программирования // современные технологии в науке и образовании-2019: материалы II международного научнотехнического форума том 4./ Рязан.гос.радиотехн.ун-т. – Рязань, 2019. – с. 180-183.

6. Пруцков А.В., Цыбулько Д.М. Применение проблемно-ориентированного объектного программирования для описания порядка работы интеллектуальных и информационных систем // Вестник Рязанского государственного радиотехнического университета. – 2014.– № 47. – С. 92-96.

7. Пруцков А.В., Цыбулько Д.М. Проблемно-ориентированный подход к пользовательскому программированию // Cloud of Science. – 2016. – Т. 3. – № 1. – С. 105-114.

УДК 004.9; ГРНТИ 20.53.19

РАЗРАБОТКА МЕХАНИЗМОВ ПРОФОРИЕНТАЦИИ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

В.С. Агейкова, А.Ю. Громов

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, viktoria0126@yandex.ru

Аннотация. В работе описан процесс профориентации с использованием методов машинного обучения для учеников старших классов. Исходная информация содержит социально-демографические, личные и академические данные от учащихся. К средствам разработки относится язык программирования R.

Ключевые слова: искусственный интеллект (ИИ), машинное обучение (ML), язык программирования R, профориентация, ассоциативные правила, классификация.

DEVELOPMENT OF MECHANISMS OF VOCATIONAL GUIDELINES USING METHODS OF MACHINE TRAINING

V.S. Ageykova, A.Y. Gromov

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, viktoria0126@yandex.ru

The summary. The paper describes the process of career guidance using machine learning methods for high school students. The source information contains socio-demographic, personal and academic data from students. The programming language R is a development tool.

Keywords: artificial intelligence (AI), machine learning (ML), programming language R, career guidance, associative rules, classification.

В настоящее время проблема эффективного образования и подготовки квалифицированных специалистов еще не решена, поскольку в ее основе лежит еще одна не менее значимая проблема - неспособность учащихся понять, чем они могут и хотят заниматься в даль-

нейшем. Профориентация – очень важный процесс, помогающий ученикам не только школ, но и вузов выбрать правильную сферу деятельности для дальнейшего обучения. В силу своего возраста, многие ученики не способны сделать правильный выбор. Собственные сомнения и незнание приводят к тому, что ученики слушают советы родственников, ориентируются на друзей, пытаются быть рациональными и выбирать будущую профессию исходя из потребностей рынка и многое другое.

Решение выявленной проблематики лежит в оказании эффективной помощи старшеклассникам, цель которой – поддержать учащихся в процессе выбора специальности и развития ключевых навыков, требуемых для обучения профессии. Существует множество тестов, которые направлены на определение склонностей учащихся к той или иной области образования. Однако все эти тесты не дают полной, а главное правдоподобной, картины для учащегося.

Но с развитием искусственного интеллекта появилась возможность сделать процесс профориентации более точным. Интеллектуальный анализ данных способен находить закономерности и связи из огромного объема необработнной информации, которые человек не смог бы заметить самостоятельно. Алгоритмы способны учитывать не только характеристики конкретного ученика, но и, опираясь на данные других учеников, предлагать более точные и верные решения.

Результатом процесса профориентации должна быть рекомендация с одним из профилей в обучении, установленными с 2019 года ФГОС среднего общего образования:

1. естественно-научный;
2. гуманитарный;
3. социально-экономический;
4. технологический;
5. универсальный.

Механизм профориентации с использованием методов машинного обучения содержит следующие этапы, изображенные на рисунке 1:

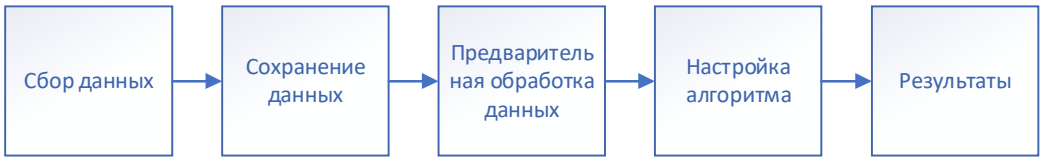


Рис. 1. Этапы процесса профориентации

Рассмотрим подробнее необходимые этапы для достижения поставленного результата.

Сбор данных

Для начала требуется определить список вопросов, которые будут заданы учащимся. Необходимо учитывать, что оценки по предметам – это малая часть интересующей информации. Помимо академических успехов учеников, интересуют их личные качества и предпочтения, демографические и социальные данные. Например, образование отца и матери, их работа, состав семьи, наличие внешкольных дополнительных занятий, состояние здоровья и многое другое.

Реализация получения информации от учащихся будет вестись за счет использования Google Форм. Это бесплатный онлайн-сервис для создания форм обратной связи, опросов, тестов и другого. Форма представляет собой отдельную веб-страницу с различными блоками, в нашем случае - это блок «вопросов». Есть функционал самостоятельно устанавливать

варианты ответов (один из предложенных, несколько из предложенных, собственный вариант).

Доступ к Google Форме осуществляется несколькими способами:

1. отправить форму по электронной почте;
2. поделиться ссылкой на форму;
3. опубликовать форму в социальных сетях;
4. встроить форму на сайт или блог.

Большой выбор вариантов отправки форм респондентам – это дополнительный плюс, поскольку позволяет охватить более широкую аудиторию и получить больше данных для дальнейшей работы.

Легкость реализации, простота использования, централизованный сбор ответов и бесплатное обслуживание – именно поэтому Google Формы выбраны для этапа сбора данных.

Сохранение данных

Сервис Google Формы при заполнении опросника респондентами автоматически создает Google Таблицы, куда записываются все полученные данные.

Google Sheets – это бесплатное онлайн-приложение, с помощью которого создаются и редактируются таблицы. Автоматическая интеграция с Google Формами и возможность работы и отслеживания данных в реальном времени делает это приложение наиболее подходящим для своих целей. Присутствует функционал многопользовательской работы с таблицей в режиме онлайн, при этом всегда доступны предыдущие версии. При этом не нарушается конфиденциальность данных, поскольку получить доступ к таблице можно либо в том случае, когда автор предоставляет доступ определенному кругу людей через электронную почту, либо предоставляет ссылку на таблицу. Для каждого из способов задаются права доступа (только для чтения или для чтения и редактирования).

Так же существует возможность выгрузить данные в удобном формате. Например, в файл excel или csv.

Предварительная обработка данных

Очистка данных – это процесс предварительной подготовки данных, который исключает из исходного набора факторы, снижающие качество данных и мешающие дальнейшему корректному анализу.

Процесс очистки данных – это очень важный этап аналитической работы, от правильности которого зависит актуальность и достоверность результатов анализа и точность построенных моделей.

Типичные факторы, которые влияют на качество данных:

- несогласованность (данные содержат конфликтующие между собой записи или расхождения)
- шум (данные содержат ошибочные записи или выбросы)
- неполнота (данные не содержат атрибутов или в них пропущены значения).

В большинстве случаев предварительная подготовка данных производится в аналитическом приложении, поскольку такие приложения имеют большой функционал для выявления и устранения факторов искажения исходных данных. Для обработки данных планируется использовать язык программирования R с открытым исходным кодом и среду разработки RStudio.

Чтобы не терять количество данных о респондентах, все пропущенные значения и поля, содержащие выбросы и ошибки будут заменены на медианные значения по атрибуту из всего исходного набора.

Далее данные будут перекодированы и приведены к числовому формату, который будет поддерживаться алгоритмами.

Настройка алгоритмов

В качестве модели для получения результатов выбрана модель классификации. Подробнее этап настройки алгоритма представлен на рисунке 2.

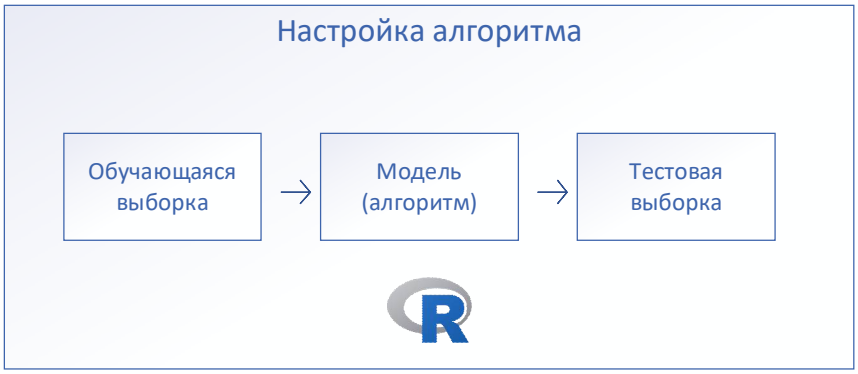


Рис. 2. Этап «Настройка алгоритма»

Реализация предполагает использование пакета «Caret», который имеет гибкую настройку. Пакет содержит набор множества функций, реализующих полный цикл разработки предсказательных моделей, включая оценку важности переменных. Выборка делится на обучающую, которая составляет 75% от всего исходного набора, и тестовую - оставшиеся 25%.

В качестве дополнительного анализа используется метод обучения обнаружения связей между переменными, который позволит определить наиболее важные правила и переменные в исходном наборе данных, которые имеют большее влияние на результат. Этот метод называется методом поиска ассоциативных правил, реализуемые с помощью алгоритма Apriori. Для этого используется пакет «arules» [1].

Результаты

Полученная модель должна удовлетворять критериям качества:

1. доля правильных ответов – accuracy;
2. метрика точности – precision;
3. метрика полноты – recall;
4. F-меры.

Если все показатели имеют высокие значения – то модель можно считать верной.

Результаты работы алгоритма предсталяют собой таблицу, где результирующим столбцом будет является столбец с номером, каждый из которых – это определенная область в образовании, представленных выше. При необходимости – можно записать данные в excel или csv файлы.

1. Шитиков В. К., Мастицкий С. Э., Классификация, регрессия, алгоритмы Data Mining с использованием R. - 2017

УДК 528.9:911.52:57.2

РАСПОЗНАВАНИЕ ЛЕСНЫХ МАССИВОВ НА СПУТНИКОВЫХ СНИМКАХ В РЕЖИМЕ РЕАЛЬНОГО ВРЕМЕНИ С ИСПОЛЬЗОВАНИЕМ НЕЙРОННЫХ СЕТЕЙ

Бакамбис Н.И.[1], Никифоров М.Б.[2], Тарасов А.С.[2]

[1]Рязанский государственный радиотехнический университет им. В. Ф. Уткина, Россия, Рязань (Конго, Браззавиль)

[2]Рязанский государственный радиотехнический университет им. В. Ф. Уткина, Россия, Рязань

Аннотация. В статье рассматриваются геоинформационные нейросетевые алгоритмы и их применение для распознавания лесных массивов на спутниковых снимках в режиме реального времени.

Ключевые слова: спутниковые снимки, геоинформационные технологии, нейросетевые алгоритмы.

FOREST RECOGNITION ON SATELLITE PICTURES IN REAL TIME USING NEURAL NETWORKS

Bakambis N.I. [1], Nikiforov M.B. [2], Tarasov A.S. [2]

[1]Ryazan State Radio Engineering University named after V.F. Utkina, Russia, Ryazan (Congo, Brazzaville)

[2]Ryazan State Radio Engineering University named after V.F. Utkina, Russia, Ryazan

Annotation. The article discusses the geographic information neural network algorithms and their application for the recognition of forests on satellite images in real time.

Keywords: Satellite imagery, geographic information technology, neural network algorithms.

Введение

Одной из актуальных задач технического зрения является автоматическая обработка спутниковых снимков в условиях, приближенных к условиям реального времени [1, 2]. В настоящее время, подобные задачи находят применение в анализе использования земельных ресурсов, для поддержания городских геодезических данных в актуальном состоянии, в частности для контроля качества и автоматического обновления геодезических данных.

В статье описывается применение сегментационных нейронных сетей для анализа спутниковых снимков с целью выделения лесных массивов. Целью проекта является распознавание верхушек деревьев на спутниковых изображениях высокого разрешения.

Подготовка данных

Для обучения нейронной сети были подготовлены более 100 изображений размером 256x256 пикселей, содержащих фрагменты спутниковых снимков. В целях повышения точности работы в различных условиях, были подготовлены данные с различных сенсоров, при различной освещенности и для различных странах. Каждое изображение размечалось на 2 класса: лес и другое.

Рис. 1. Пример размеченного изображения

Полученная выборка применялась для обучения нейронной сети UNet [3]. В качестве платформы для обучения применялись TensorFlow/Keras.

Обучение

Обучение нейронной сети производилось с использованием бесплатного сервиса Google Colab, предоставляющий ресурсы кластера GPU для ускорения процесса обучения.

```
[5]  1
     2 model.fit(x=X_train2, y=Y_train2,batch_size=3,epochs=1000)

Epoch 148/1000
55/55 [==============================] - 0s 8ms/step - loss: 0.1202 - mean_iou: 0.7963
Epoch 149/1000
55/55 [==============================] - 0s 8ms/step - loss: 0.1401 - mean_iou: 0.7967
Epoch 150/1000
55/55 [==============================] - 0s 8ms/step - loss: 0.1553 - mean_iou: 0.7972
Epoch 151/1000
55/55 [==============================] - 0s 8ms/step - loss: 0.1403 - mean_iou: 0.7974
Epoch 152/1000
55/55 [==============================] - 0s 8ms/step - loss: 0.1261 - mean_iou: 0.7979
Epoch 153/1000
55/55 [==============================] - 0s 8ms/step - loss: 0.1261 - mean_iou: 0.7984
```

Рис. 2. Процесс обучения

В результате, точность работы сети составила 89,5%.

Применение

Для анализа полученных результатов, были подготовлены спутниковые снимки 6 различных городов, после чего, каждый из них был обработан при помощи данной нейросети.

В результате были получены результаты, представленные в таблице 1.

Таблица 1. Площадь зеленых зон для различных городов

Город	Площадь
Рязань	21,55 %
Владимир	23,18 %
Тула	16,63 %
Прага	33,90 %
Москва	55,21%

Основываясь на данном показателе можно сделать выводы о количестве парков и зеленых насаждений в городе. Кроме того, с использованием полученных результатов можно составить тепловую карту, демонстрирующую недостаток зеленых насаждений на территории города:

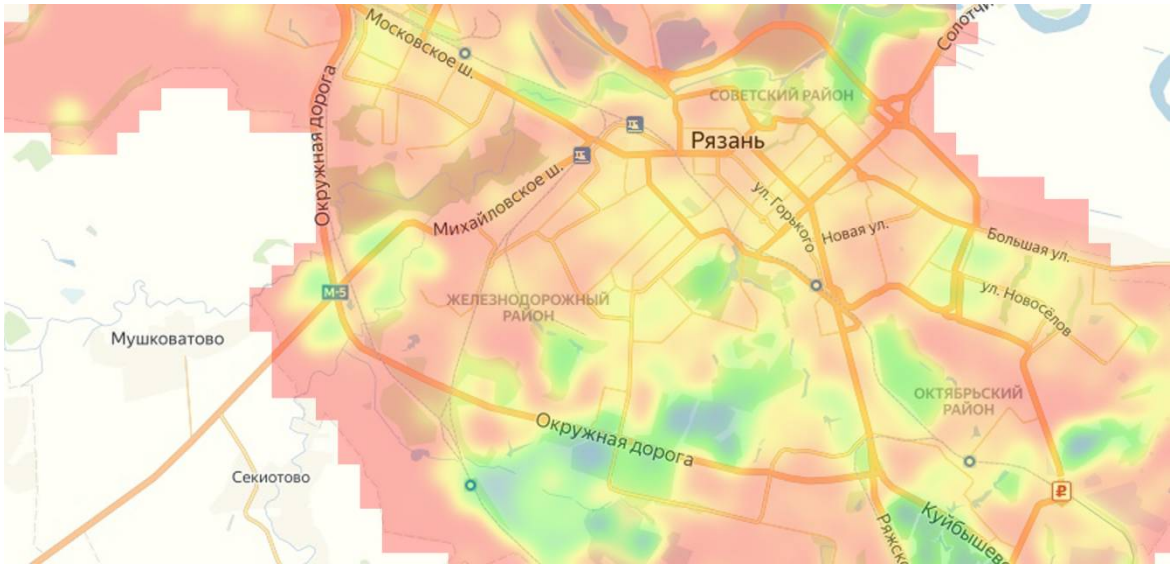


Рис. 3. Тепловая карта зеленых зон в пределах территории города

Красным выделены участки, для которых количество зеленых зон недостаточное.

Выводы

Разработанный алгоритм позволяет с высокой точностью классифицировать спутниковые снимки и определять наличие на них зеленых зон. Данный подход в дальнейшем может помочь в улучшении экологической ситуации, а также помочь оперативно определить незаконные вырубки леса.

Библиографический список

1. Badrinarayanan Vijay, Kendall Alex, Cipolla Roberto. Segnet: A deep convolutional encoder-decoder architecture for image segmentation // IEEE transactions on pattern analysis and machine intelligence. — 2017. — Vol. 39, no. 12. — P. 2481–2495.

2. Brostow Gabriel J., Fauqueur Julien, Cipolla Roberto. Semantic Object Classes in Video: A High-Definition Ground Truth Database // Pattern Recognition Letters. — 2008.

3. Ronneberger Olaf, Fischer Philipp, Brox Thomas. U-Net: Convolutional Networks for Biomedical Image Segmentation // CoRR. — 2015. — Vol. abs/1505.04597. — 1505.04597.

3. Акинин М.В., Акинина Н.В., Никифоров М.Б. Интеллектуальные алгоритмы построения двумерных карт местности: монография. – М.: КУРС, 2020. – 96 с.

4.Акинин М.В., Никифоров М.Б., Таганов А.И. Нейросетевые системы искусственного интеллекта в задачах обработки изображений: монография. – М.: Науно-техническое издательство «Горячая линия — Телеком», 2016. – 152 с.

5. Обработка изображений в авиационных системах технического зрения /Под ред. Л.Н. Костяшкина, М.Б. Никифорова. – М.: ФИЗМАТЛИТ, 2016. – 240 с.

УДК 004.932.2, 159.942.33; ГРНТИ 28.23.15, 15.01.77

ИСПОЛЬЗОВАНИЕ АЛГОРИТМА ТРИАНГУЛЯЦИИ ДЕЛОНЕ ДЛЯ АНАЛИЗА МИКРОВЫРАЖЕНИЙ ЛИЦА

В.А. Саблина , С.В. Гришин

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Россия, Рязань, sablina.v.a@evm.rsreu.ru

Аннотация. Современные технологии дают возможность обнаруживать и распознавать не только лицо, но и отдельные возникающие на нем микровыражения, анализируя последовательность изображений. В статье рассмотрен подход к построению маски лица на основе триангуляции Делоне. Показаны сравнительные особенности проявления семи базовых эмоций на лице человека. Построены маски на основе алгоритма триангуляции Делоне для каждой из базовых эмоций. Предложено использовать построенные маски для анализа микровыражений лица.

Ключевые слова: анализ микровыражений лица, базовые эмоции, триангуляция Делоне, антропометрические точки лица, маска лица.

USING OF THE DELAUNAY TRIANGULATION ALGORITHM FOR FACIAL MICRO-EXPRESSIONS ANALYSIS

V.A. Sablina , S.V. Grishin

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, sablina.v.a@evm.rsreu.ru

Abstract. The modern technologies make it possible to spot and recognize not only the face but also the certain micro-expressions arising on it by analyzing the image sequence. In the paper an approach to the facial mask construction on the basis of the Delaunay triangulation is considered. The comparative characteristics of the manifestation of the seven basic emotions on the human face are shown. The masks for each of the basic emotions are constructed on the basis of the Delaunay triangulation. The constructed masks are proposed to use for facial micro-expression analysis.

Keywords: facial micro-expressions analysis, basic emotion, Delaunay triangulation, facial landmarks, facial mask.

1. Введение

Эмоции человека всегда отражаются на его лице [1]. По лицу можно определить, соответствуют ли истинные эмоции тому, что человек говорит. Таким образом, можно узнать правду. Ведь лицо постоянно меняется и служит каналом для передачи информации. Эмоции и интенсивность их выражения могут изменяться, но всё это проявляется на лице. Можно заметить стремление скрыть эмоции. Если человек (сознательно или нет) пытается не проявлять эмоции на лице, всё равно возникают кратковременные выражения, называемые микровыражениями, по которым можно определить испытываемые человеком в данный момент эмоции [2].

2. Микровыражения и базовые эмоции человека

Микровыражение – это короткое непроизвольное выражение лица, появляющееся на лице человека, пытающегося скрыть или подавить эмоцию. Обычно микровыражения длятся от 1/25 до 1/5 секунд [3]. Это быстрые, интенсивные выражения скрытых эмоций. Таким образом, главные характеристики микровыражений – моментальность и непроизвольность. Моментальность – микровыражения могут появиться и исчезнуть с лица в доли секунды. То есть, они появляютсяна столько стремительно, что многие люди просто не замечают их. Непроизвольность – микровыражения вызваны непроизвольными движениями мышцами лица. Большинство людей не в состоянии контролировать мышцы, которыми выражаются эмоции.

Микровыражения отражают на лице проявления тех же эмоций, что и обычные макровыражения. Согласно Полу Экману, выделяют семь базовых эмоций человека: счастье (рис. 1, а), печаль (рис. 1, б), гнев (рис. 1, в), отвращение (рис. 1, г), страх (рис. 1, д), презрение (рис. 1, е) и удивление (рис. 1, ж) [4].

За последнее десятилетие появилось большое количество методов и алгоритмов для автоматического анализа микровыражений лица. Ранее анализ осуществлялся без привлечения технических средств преимущественно профессиональными психологами. В настоящее время исследования стали возможны благодаря созданию специальных наборов изображений, отражающих спонтанные эмоции и снятых высокоскоростными камерами. Однако до сих пор автоматическая классификация эмоций по микровыражениям является практически очень сложной задачей. Наиболее распространены подходы, основанные на предварительном выявлении движений мышц лица, как объективных событий, и только затем по ним распознаются непосредственно сами эмоции [5].

Рис. 1. Основные эмоции: а – счастье; б – печаль; в – гнев; г – отвращение; д – страх; е – презрение; ж – удивление

3. Триангуляции Делоне в задачах анализа микровыражений

Для решения задач анализа микровыражений используют многоэтапные конвейеры алгоритмов, в совокупности позволяющих обнаруживать и распознавать микровыражения [6]. Важной подзадачей является построение маски лица, описывающей основные области на лице человека. Далее рассмотрен подход к ее решению на основе использования триангуляции Делоне. Это позволит нивелировать различия лиц конкретных людей, мешающие их сравнению со среднестатистическим эталоном [7].

Триангуляция Делоне множества точек – это триангуляция, являющаяся подразбиением Делоне. А подразбиение Делоне множества точек – такое разбиение выпуклой оболочки множества точек на множество выпуклых фигур, что в окружности, описанной вокруг любой из фигур, не находится никаких точек из множества [8]. На рисунке 2 наглядно проиллюстрировано выполнение указанного условия. Впервые задача построение подобной триангуляции была поставлена советским математиком Борисом Николаевичем Делоне в 1934 году.

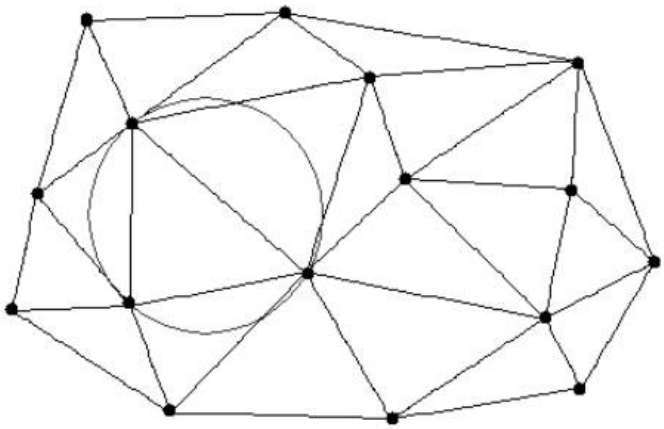
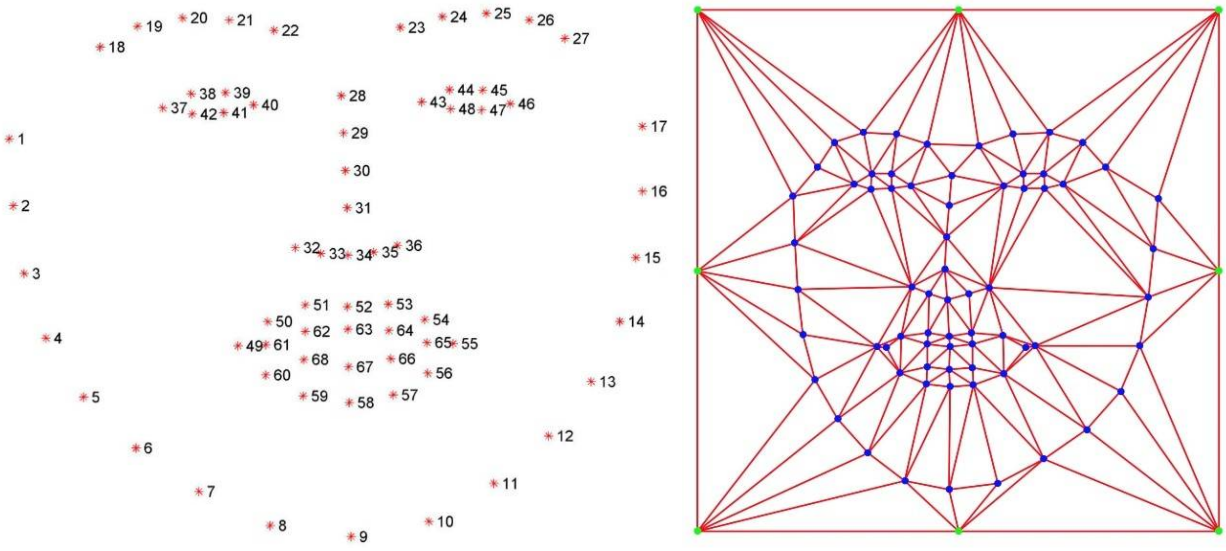


Рис. 2. Выполнение условия триангуляции Делоне

4. Этапы технологии построения маски лица

Технологию построения маски лица на основе триангуляции Делоне можно реализовать в несколько этапов. На первом этапе определяют местонахождение лица на изображении, а затем и его основных подобластей [9]. После этого необходимо обнаружить антропометрические точки лица (рис. 3, а). Потом найденные точки соединяются с помощью алгоритма триангуляции Делоне. Полученные ключевые контуры представляют собой маску лица (рис. 3, б). В результате можно описать следующие основные элементы: само лицо, левый глаз, правый глаз, левая бровь, правая бровь, левый зрачок, правый зрачок, нос, губы.



а б

Рис. 3. Этапы технологии маскирования: а – поиск антропометрических точек; б – поиск контуров лица

5. Обнаружение и распознавание микровыражений с помощью маски

Для обнаружения и распознавания микровыражений необходимо задать эталонную маску – нейтральное выражение лица. По отношению к этому эталону производятся измерения отклонений значений положения точек мимически активных частей лица. Пример эталонной маски для нейтрального выражения лица показан на рисунке 4.

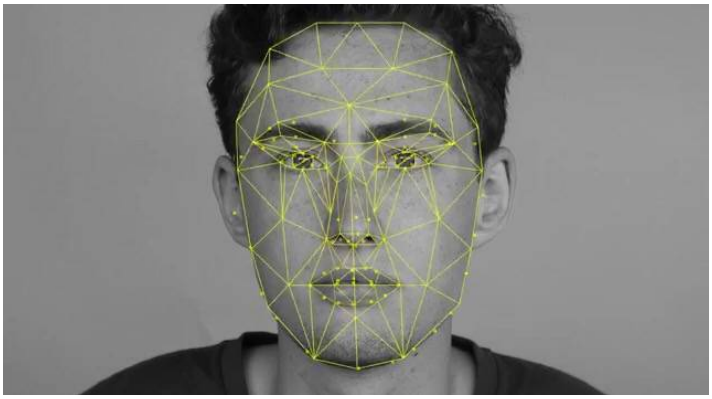


Рис. 4. Эталонная маска для нейтрального выражения лица

Для микровыражения характерны кратковременность проявления с последующим возвратом к нейтральному выражению лица. Для каждой эмоции характерны определённые движения лица. Основная задача анализа микровыражений – определить на основе кратковременных изменений выражения лица, какую эмоцию испытывал человек. При использовании высокоскоростной камеры можно получить последовательность кадров, на которых можно обнаружить, что мимика лица меняется в соответствии с некоторой эмоцией, которая вызывает микровыражение. Такие события, как изменение положения антропометрических точек рта, глаз или других элементов лица фиксируются, таким образом, можно обнаружить начало микровыражения в последовательности кадров. По истечении заданного временного промежутка, характерного для микровыражения, положение антропометрических точек лица сравнивается с эталонным. Если точки вернулись в положение, характерное для нейтрального выражения лица, то микровыражение завершилось. После обнаружения выполняется распознавание микровыражения. Для этого анализируются движения отдельных точек во время микровыражения и их отклонения относительно нейтрального положения, тем самым определяются движения бровей, глаз, рта. Распознавание микровыражения возможно путем сравнения движения антропометрических точек различных элементов лица с известными типовыми движениями мышц лица. По совокупности этих движениям можно распознать эмоцию из семи базовых, которая, скорее всего, послужила причиной возникновения анализируемого микровыражения.

В результате проведенного исследования построены маски на основе триангуляции Делоне для семи базовых эмоций, характеризующие выражение каждой эмоции в момент наибольшего отклонения от нейтрального выражения лица. Построенные маски описывают легко различимые лицевые движения для базовых эмоций. Далее на рис. 5-8 элементы лица для нейтрального выражения показаны зеленым цветом, а для рассматриваемой эмоции – красным.

Для эмоции счастья характерны приподнятые щёки и движения мышц вокруг глаз. На рис. 5 представлена маска для эмоции счастья на фоне нейтрального выражения лица.

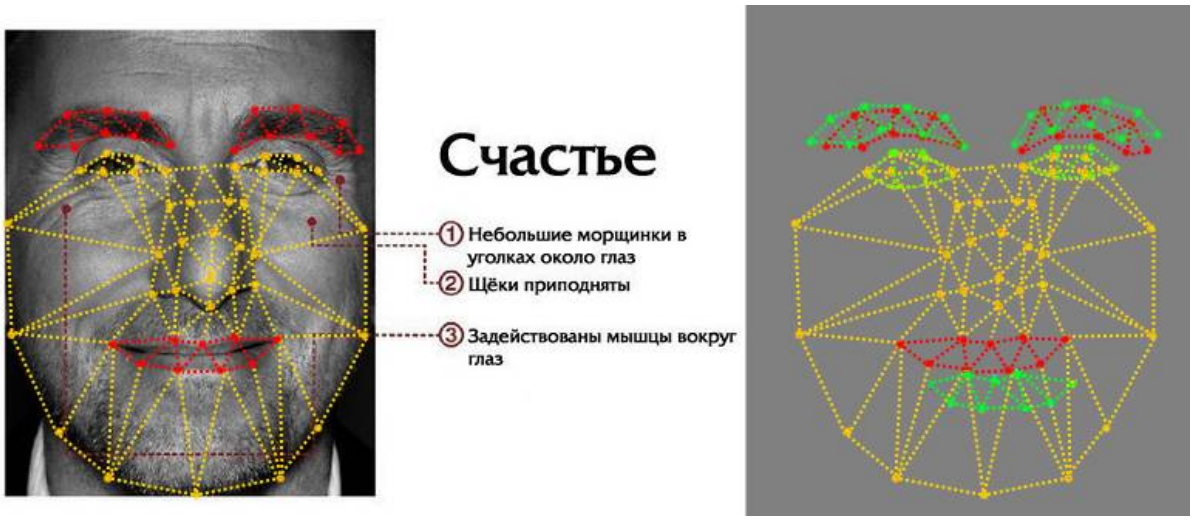
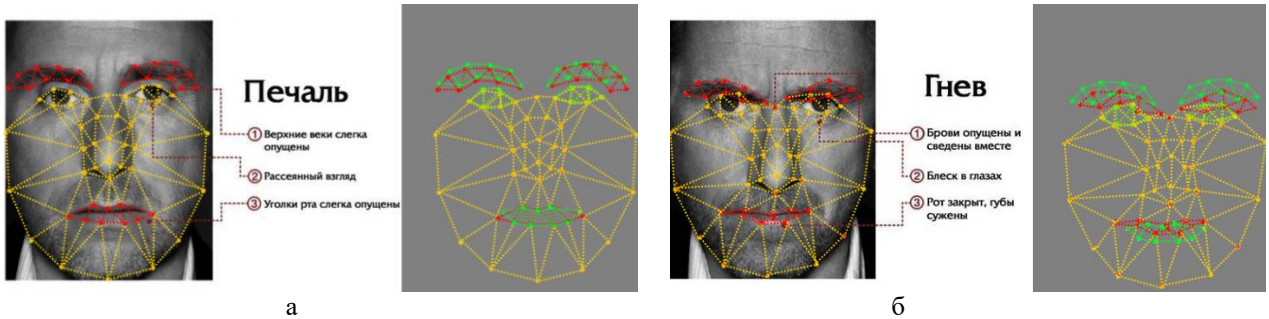


Рис. 5. Результат построения маски для эмоции «счастье»

Для эмоции печали характерны слегка опущенные верхние веки и слегка опущенные уголки рта. На рис. 6, а представлена маска для эмоции печали на фоне нейтрального выражения лица.

Для эмоции гнева характерны опущенные и сдвинутые к переносице брови, сжатые губы. На рис. 6, б представлена маска для эмоции гнева на фоне нейтрального выражения лица.

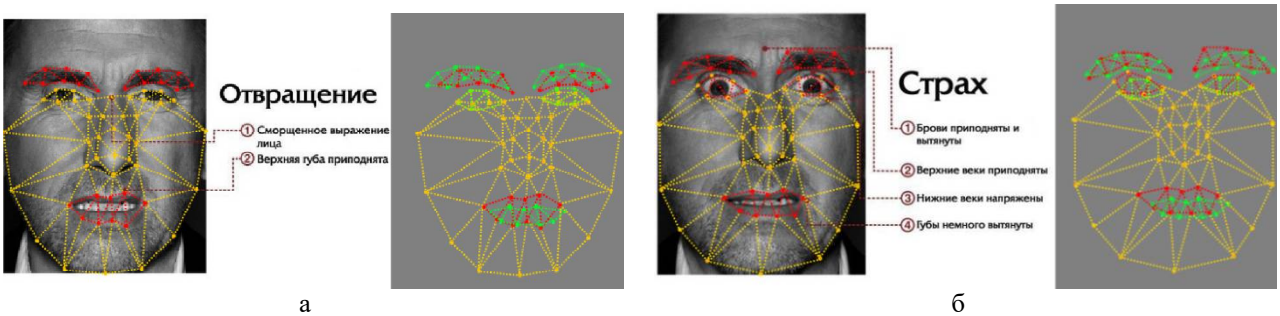


а б

Рис. 6. Результаты построения масок для эмоций: а – «печаль»; б – «гнев»

Для эмоции отвращения характерны сморщенное выражение лица и приподнятая верхняя губа. На рисунке 7, а представлена маска для эмоции отвращения на фоне нейтрального выражения лица.

Для эмоции страха характерны приподнятые и вытянутые брови, напряжённые нижние веки и немного вытянутые губы. На рисунке 7, б представлена маска для эмоции страха на фоне нейтрального выражения лица.

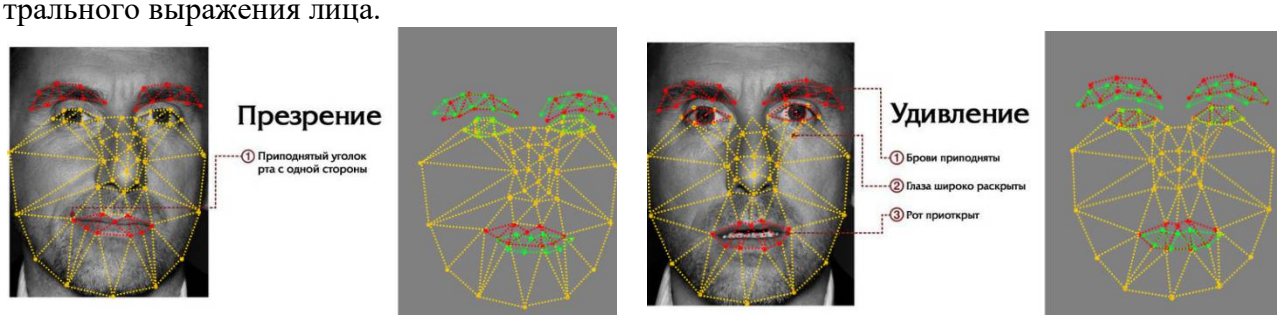


а б

Рис. 7. Результаты построения масок для эмоций: а – «отвращение»; б – «страх»

Для эмоции презрения характерен приподнятый уголок рта с одной стороны. На рис. 8, а представлена маска для эмоции презрения на фоне нейтрального выражения лица.

Для эмоции удивления характерны приподнятые брови, широко раскрытые глаза, приоткрытый рот. На рисунке 8, б представлена маска для эмоции удивления на фоне нейтрального выражения лица.



а б

Рис. 8. Результаты построения масок для эмоций: а – «презрение»; б – «удивление».

6. Заключение

Технологии компьютерного зрения в области обнаружения и распознавания микровыражений лица продолжают развиваться и представляют собой перспективное направление для дальнейших исследований. Микровыражения содержат значительное количество информации об истинных эмоциях человека, которая может применяться работниками таможни, полиции и пограничной службы, специалистами по кадрам, журналистами во время интервью и людьми, которым часто приходится искать обман или подтверждать факты. Построенные маски на основе алгоритма триангуляции Делоне для семи базовых эмоций могут быть использованы при разработке и реализации технологий автоматического анализа микровыражений лица человека.

Библиографический список

1. Paul Ekman, Emotion in the Human Face, 2nd Edition, Malor Books, 2013, 456 p.

2. Профайлинг с Анной Кулик. Микровыражения – утечки лица, верный признак лжи говорящего [Электронный ресурс]. – URL: https://anna-kulik.ru/135 (дата обращения: 31.03.2020).

3. Википедия. Микровыражение [Электронный ресурс]. – URL: https://ru.wikipedia.org/wiki/Микровыражение (дата обращения: 31.03.2020).

4. Theory and Practice. Стать доктором Лайтманом: как научиться распознавать ложь [Электронный ресурс]. – URL: https://theoryandpractice.ru/posts/10337-lie-to-me (дата обращения: 31.03.2020).

5. Yee-Hui Oh, John See, Anh Cat Le Ngo, Raphael C.-W. Phan, and Vishnu M. Baskaran, “A Survey of Automatic Facial Micro-Expression Analysis: Databases, Methods, and Challenges,” Frontiers in Psychology Journal, Volume 9, Article 1128, 2018, 21 p.

6. Саблина В.А., Сергеева А.Д. Методы распознавания микровыражений лица: обзор // Современные технологии в науке и образовании – СТНО-2019: сб. тр. II междунар. науч.-техн. форума: в 10 т. Т.4./ под общ. ред. О.В. Миловзорова. – Рязань: Рязан. гос. радиотехн. ун-т, 2019. – С. 117-122.

7. Zhaoyu Lu, Ziqi Luo, Huicheng Zheng, Jikai Chen, and Weihong Li, “A Delaunay-Based Temporal Coding Model for Micro-expression Recognition,” in Proceedings of Asian Conference on Computer Vision (ACCV), 2014, pp. 698-711.

8. Университет ИТМО. Триангуляция Делоне [Электронный ресурс]. – URL: https://neerc.ifmo.ru/wiki/index.php?title=Триангуляция_Делоне (дата обращения: 31.03.2020).

9. Anna D. Sergeeva, Alexander V. Savin, Victoria A. Sablina, and Olga V. Melnik, Emotion Recognition from Micro-Expressions: Search for the Face and Eyes, 8th Mediterranean Conference on Embedded Computing (MECO) Proceedings. Budva, Montenegro, 2019, pp. 632-635.

УДК 004.318; ГРНТИ 50.09.33

ВЛИЯНИЕ КЭШ – ПАМЯТИ НА ВЫЧИСЛИТЕЛЬНУЮ МОЩНОСТЬ ПРОЦЕССОРА

А.С. Чалов, М.Б. Никифоров
Рязанский государственный радиотехнический университет имени В.Ф. Уткина,
Россия, Рязань, nikiforov.m.b@evm.rsreu.ru

Аннотация. Рассматривается влияние на производительность вычислительной системы с процессором кэш-памяти на примере различных алгоритмов и программ.
Ключевые слова: кэш-память, производительность вычислительной системы, процессор.

IMPACT OF CACHE MEMORY ON PROCESSOR PROCESSING POWER

A.S. Chalov, M.B. Nikiforov
Ryazan State Radio Engineering University named after V.F. Utkin,
Russia, Ryazan, nikiforov.m.b@evm.rsreu.ru

Abstract. How the performance of a computer system with a cache processor is affected by various algorithms and programs.
Keywords: Cache, computing performance, processor.

Память как человеческий мозг. Он используется для хранения «данных» и «инструкций». Компьютерная память - это пространство хранения в компьютере, где должны обрабатываться данные и храниться инструкции, необходимые для обработки. Память делится на большое количество мелких частей, называемых ячейками. Каждое местоположение или ячейка имеет уникальный адрес, который варьируется от нуля до объема памяти минус один. Со временем технологии эволюционируют, расширяется функционал и имеющиеся оперативной памяти не хватает для достижения необходимой вычислительной мощности ЭВМ, поэтому появляется кэш – память.

Кэш-память — это компьютерный компонент, который повышает эффективность извлечения данных из памяти компьютера. Он действует как область временного хранения, из которой процессор компьютера может получать данные. Эта область временного хранения, известная как кэш-память, более доступна для процессора, чем основной источник памяти компьютера, обычно в виде DRAM.

Кэш-память чаще является частью ЦП (центрального процессора), поскольку она обычно интегрируется непосредственно в микросхему ЦП или размещается на отдельной микросхеме, которая имеет отдельную шину, соединенную с ЦП. Следовательно, она более доступна для процессора и способна повысить эффективность, поскольку она физически близка к процессору. Скорость процессора значительно увеличивается по сравнению с величиной задержки доступа к основной памяти. Эффект этого разрыва может быть уменьшен путем эффективного использования кэш-памяти.

Наиболее часто используемые данные или инструкции хранятся в кэше, поэтому к ним можно получить доступ с очень высокой скоростью, что повышает общую производительность компьютера. Существует несколько уровней кэша, причем последний уровень самый большой и самый медленный, а первый уровень самый быстрый и самый маленький.

В большинстве процессоров кэш первого уровня (L1) находится в процессоре, а кэш второго уровня (L2) и кэш третьего уровня (L3) находятся на отдельной микросхеме. В многоядерных процессорах каждый процессор имеет свой собственный кэш L1. Тактовая частота процессора в несколько сотен раз выше, чем задержка доступа к основной памяти, поэтому кэш-память обеспечивает сокращение этого разрыва и повышение производительности системы. Структура многоядерного процессора представлена на рисунке 1.

Если кратко описать процесс работы кэш-памяти, то данные поступают из ОЗУ в кэш L3, затем в L2 и, наконец, в L1. Когда процессор ищет данные для выполнения операции, он

сначала пытается найти их в кэше L1. Если процессор может их найти, то условие называется попаданием в кэш, если не находит, то ищет их в L2, а затем в L3.Если процессор не находит данные, он пытается получить к ним доступ из основной памяти. Это называется промахом кэша (Cache Miss).

Время, необходимое для доступа к данным из памяти, называется задержкой. L1 имеет самую низкую задержку, будучи самой быстрой и ближайшей к ядру, а L3 имеет самую высокую. Задержка увеличивается, когда происходит промах кэша, поскольку процессор ищет данные в основной памяти. Чем больше задержка, тем ниже скорость работы программы, а это, в конечном итоге, влияет на производительность системы.

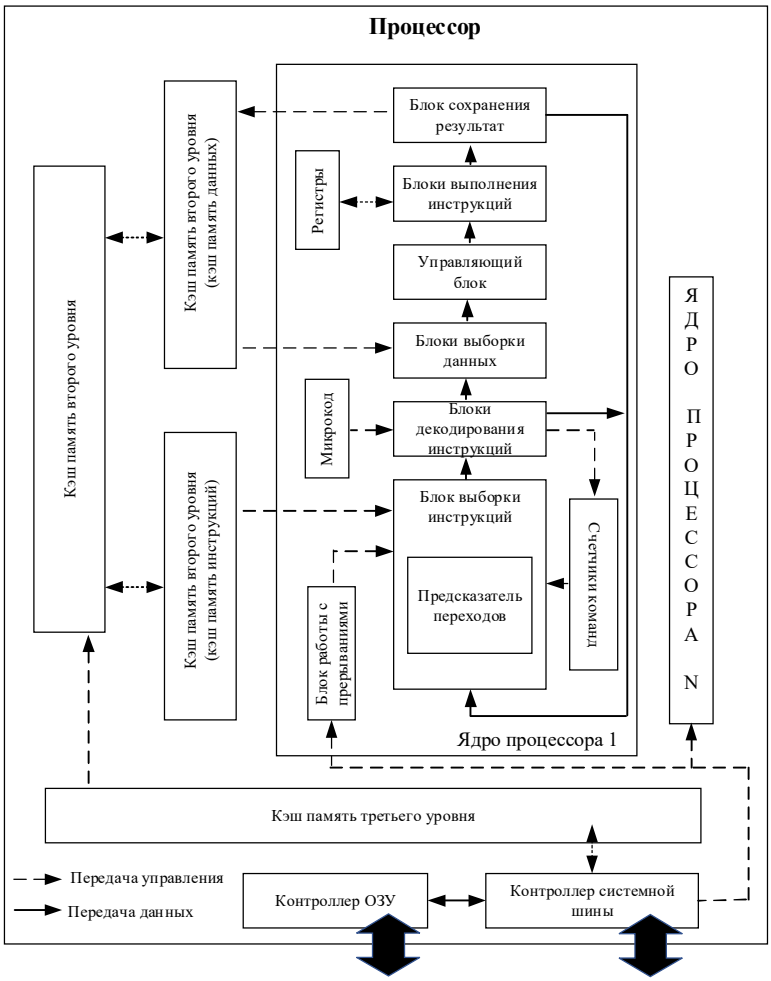


Рис. 1. Структура многоядерного процессора

Однако, любое приложение, написанное на языке высокого уровня, можно оптимизировать. Для наглядности приведён пример, в котором продемонстрирована разница доступа к массиву памяти.

```
Stopwatch stopWatch = new Stopwatch();
stopWatch.Start();

int[] arr = new int[64 * 1024 * 1024];
```

Цикл 1:

```
for (int i = 0; i < arr.Length; i++) arrr[i] *= 5;

Цикл 2:
for (int i = 0; i < arr.Length; i += 15) arr[i] *= 5;

stopWatch.Stop();
TimeSpan ts = stopWatch.Elapsed;
string elapsedTime = String.Format ("{0:00} :{1:00} :{2:00} .{3:00}", ts.Hours, ts.Minutes, ts.Seconds, ts.Milliseconds / 10);
```

Результат:
С первым циклом: 41мс
Со вторым циклом: 19 мс

В первом цикле умножается каждое число массива на 5, а во втором только каждое 15 число массива. Т.е. в первом случае мы обрабатываем массив последовательно, увеличивая счётчик на 1, а во втором случае перескакиваем, увеличивая счётчик на 15. Соответственно второй цикл выполняет лишь 5-8% работы первого цикла, однако разница между первым циклом и вторым – несущественная. Причина, по которой циклы занимают, фактически, одинаковое количество времени, связана с памятью, а именно с доступом к массиву памяти, а не умножением чисел. Для наглядности приведены графики сравнения скорости обработки циклов в зависимости от увеличения размера массива (рисунки 2-4).

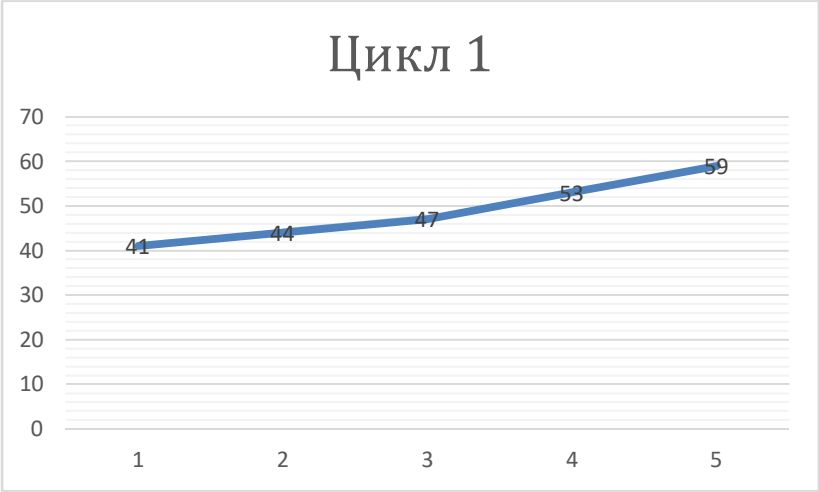


Рис. 2. Обработка цикла 1

Рис. 3. Обработка цикла 2

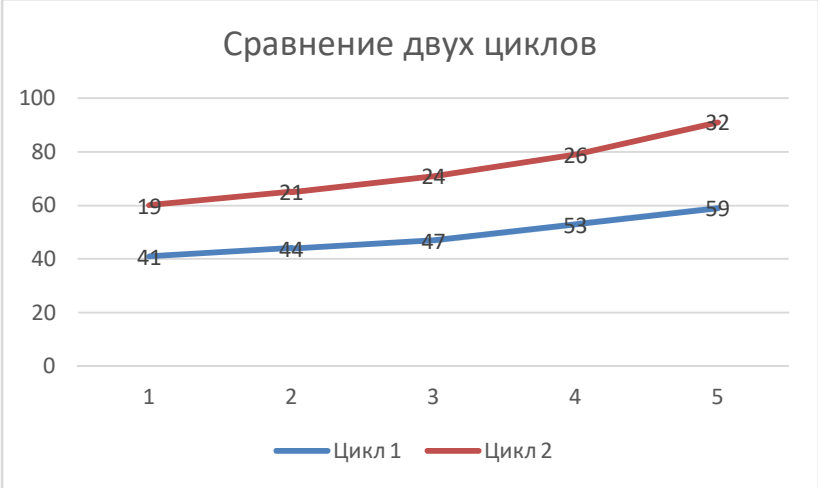


Рис. 4. Сравнение скорости обработки двух циклов

Далее продемонстрирован пример, в котором наглядно показано, как размер кэша L1, L2 и L3 влияет на скорость вычислений.

```
Stopwatch stopWatch = new Stopwatch();
stopWatch.Start();

int[] arr = new int[64 * 1024 * 1024];
int steps = 64 * 1024 * 1024;
int L = arr.Length - 1;

for (int i = 0; i < steps; i++){arr[(i * 16) & L]++; }
stopWatch.Stop();
TimeSpan ts = stopWatch.Elapsed;
string elapsedTime = String.Format ("{0:00} :{1:00} :{2:00} .{3:00}", ts.Hours, ts.Minutes, ts.Seconds, ts.Milliseconds / 10);
```

Результат:

До 9МБ: 31 – 89мс.
Свыше 9МБ: 1с 27мс и далее.

В данном случае мы обходим массив, который увеличивает каждый 16й элемент для того, чтобы сменить строку кэша, поскольку процессор не обращается к памяти напрямую, т.е. побайтно, а извлекает память поблочно по 64 байта. Эти блоки и называются строками кэш-памяти. Когда мы считываем информацию из определенной ячейки памяти, то вся строка извлекается из основной памяти в кэш. Поскольку 16 значений типа int (в C#) занимают 16*4=32 байта, что является одной строкой кэш-памяти. Так как мой процессор имеет кэш-память L3 размером в 9 МБ, то в программе после изменения размера массива, выходящего за границы 9 МБ, заметно значительное понижение скорости вычисления. Для наглядности предоставлен график сравнения размера кэш-памяти и скорости её обработки (рис. 5).

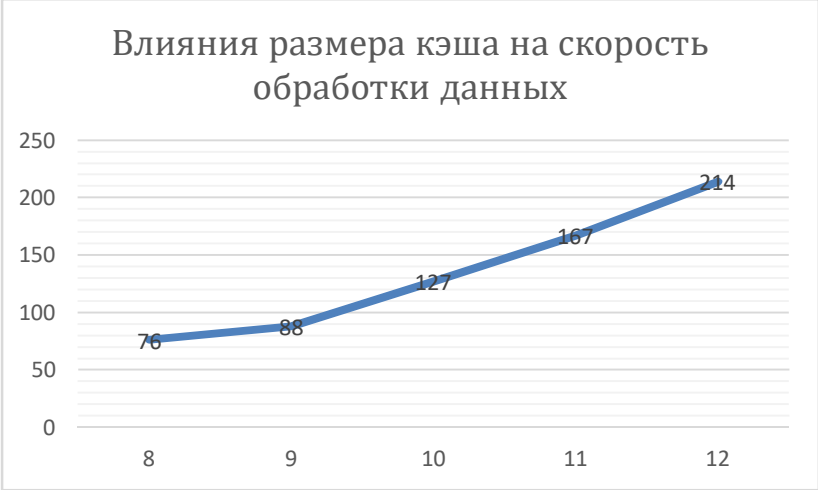


Рис. 5. График зависимости размера кэш-памяти от скорости вычисления

Таким образом, чем больше размер кэш-памяти и характеристики процессора, тем выше скорость вычислительная скорость, особенно в случае, когда приложение, написанное на языке высокого уровня, оптимизировано.

Библиографический список

1. Чалов А.С., Никифоров М.Б. Актуальные проблемы современной науки и производства – Влияниемногопоточности на производительность ЭВМ.Материалы IV Всероссийской научно-технической конференции, 2019. С. 178 – 183.
2. Мелехин, В.Ф. Вычислительные машины, системы и сети: учебник для Вузов / В.Ф. Мелехин, Е.Г.Павловский. М.: Академия. 2006. 555 с.
3. Анашкин, А.С. Техническое и программное обеспечение распределенных систем управления: Уч. пособие для Вузов / А.С.Анашкин, Э.Д.Кадыров, В.Г.Харазов; СПбГТИ(ТУ), СПбГГИ(ТУ) им. Г.В.Плеханова. СПб, 2004. 368 с.
4. Гук. М. Процессоры intel от 8086 до Pentium 4. С-Петербург -“Питер Паблишинг” – 2002.
5. Обзор процессоров и шин ПВМ. Москва – 2001.
6. Blumrich, M. et. al. Design and Analysis of the Blue Gene/L Torus Interconnection Network. IBM Research Report, 2003.
7. Hameed, R. et al. Understanding sources of inefficiency in general-purpose chips. International Symposium on Computer Architecture (2010).
8. Pollack, F. Pollack's Rule of Thumb for Microprocessor Performance and Area.

УДК 004.932; ГРНТИ 59.41.29

СПОСОБЫ ПОВЫШЕНИЯ ТОЧНОСТИ ИЗМЕРЕНИЯ ДАЛЬНОСТИ ВРЕМЯПРОЛЕТНЫМИ КАМЕРАМИ

Т.М. Савосина , Н.А. Яценя
Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, natashayatc@yandex.ru

Аннотация. В данной работе рассматриваются методы измерения дальности с помощью времяпролетных камер, причины и факторы влияющие на появление ошибок измерения дальности, а также способы повышения точности измерений дальности времяпролетными камерами.
Ключевые слова: времяпролетные камеры, методы измерение дальности, ошибки измерения дальности из-за переотражений, ошибки измерения дальности из-за поглащения, коррекция ошибок.

WAYS TO IMPROVE THE ACCURACY OF MEASURING RANGE BY TIME-OF-FLIGHT CAMERAS

T.M. Savosina, N.A. Yatcenya
Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, natashayatc@yandex.ru

Abstract. In this paper, we consider methods of measuring ranges using time-of-flight cameras, the causes and factors that influence the appearance of range-measurement errors, and also ways to improve the accuracy of ranging by time-of-flight cameras.
Keywords: time-of-flight cameras, range measurement methods, range measurement errors due to reflections, range measurement errors due to absorption, error correction.

Измерение расстояние до объектов – важная составляющая современных систем технического зрения. Существует несколько типов сенсоров, измеряющих расстояние, основываясь на разных физических принципах измерения и алгоритмах обработки сырой информации.

- Классические стереосистемы (Depth from Stereo) используют два сенсора и алгоритмы вычисления диспаритета для пикселей. Как разновидность стереосистем, системы с более чем двумя камерами.
- Камеры с структурным подсветом (Structured Light Sensor). В таких системах проектор камеры освещает сцену структурным светом, а алгоритм определяет дальность по распределению этой структуры на изображении, фиксируемом сенсором.
- Время пролетные камеры (Time of Flight – ToF) в этих камерах дальность вычисляется путем измерения задержки времени между излученным сигналом подсвета и отраженным от объекта [1].
- Камеры светового поля или пленооптические камеры (Light Field Sensor) - сенсор такой камеры спроектирован таким образом, что позволяет измерить характеристики светового поля, по которым в дальнейшем алгоритм восстанавливает изображения с возможностью изменения характеристик оптической системы уже после съемки.
- Лидары (Lidar) – работают по принципу измерения задержки времени между излученным световым сигналом и отраженным, но по механике и энергетике отличаются от Time of Flight камер.
- Радары – работают также как и лидары, но вместо светового сигнала используют излучение с длиной волны от нескольких миллиметров до нескольких десятков метров, в зависимости от назначения и конструкции.

Рассмотрим наиболее интересный и весьма новый способ измерения расстояния до объектов, набирающий популярность, а именно Time of Flight камерами.

Методы измерения дальности ToF камерами

ToF камера работает путем освещения сцены излучателем модулированного сигнала и приема отраженного света матрицей ToF сенсора. В качестве излучателя применяется диодная подсветка или прожекторы на базе полупроводниковых поверхностно-излучающих лазеров с вертикальным резонатором (VCSEL flood illuminator). В качестве приемника используют матрицу однофотонных лавинных диодов (SPAD) [1, 2].

Существует два способа определения дальности, используемых в ToF камерах: прямой (Direct-ToF) и косвенный (Inderect-ToF) [2, 3].

В способе Direct-ToF подсчитывается время между излучением и обнаружением отражения модулированного сигнала с помощью быстрого счетчика.

В Inderect-ToF способе определяется смещение фазы излучаемого и принимаемого излучения. Indirect-ToF способ в реальных устройствах часто реализуется одним из двух основных методов: импульсным (Pulsed) или методом непрерывного излучения (continuous-wave или CW).

Влияние частоты модуляции подсвета на точность измерений

Импульсный метод прост. Источник света освещается в течение короткого периода (Δt), и энергия отраженного излучения накапливается в виде заряда в каждом «пикселе» ToF сенсора в двух емкостях, в каждую из которых заряд переносится за половину периода модулированного сигнала.Таким образом, заряды, накопленные за время после оцифровки составят значения Q1 и Q2. Расстояние *d* до объектов, отожествляемых пикселями, рассчитывается по формуле:

$$d = \frac{c\Delta t}{2}\left(\frac{Q_2}{Q_1 - Q_2}\right),$$

где c – скорость света.

В отличие от первого, CW метод накапливает заряды за полупериоды со сдвигом фазы шагом на 90 градусов, делая всего четыре замера.

Имея четыре значения Q_1, Q_2, Q_3 и Q_4 принятого сигнала, можно вычислить сдвиг фазы принятого излучения по формуле

$$\Delta\varphi = arctan\left(\frac{Q_3 - Q_4}{Q_1 - Q_2}\right),$$

а дальность *d* (depthmap) - по формуле $d = \frac{c}{4\pi f}\Delta\varphi$ [1, 4].

Также доолнительно появляется возможность оценить интенсивность яркости объектов (A) и смещение излучения (B) по формулам:

$$A = \frac{\sqrt{(Q_1 - Q_2)^2 + (Q_3 - Q_4)^2}}{2} \text{ и } B = \frac{Q_1 + Q_2 + Q_3 + Q_4}{4}.$$

При этом оценка СКО измеренных значений составит:

$$\sigma = \frac{c\sqrt{A+B}}{4\sqrt{2}\pi f c_d A},$$

где c_d – контраст модуляции.

Ошибки измерения дальности из-за поглощения ИК излучения подсвета

При использовании ToF наблюдаются следующие ошибки измерения дальности:

- Из-за поглощения некоторыми материалами ИК излучения подсвета Q_1, Q_2, Q_3 и Q_4 (см. формулу в п.2) измененные ToF сенсором будут иметь малые значения. Помимо этого, эти величины будут приблизительно равными и в сумме не сильно отличаться от значений (B);
- Малость величины значений не позволит вычислить угол смещения фазы отраженного модулирования освещения с приземлённой точностью;
- Наличие шума в значениях Q1-Q4 увеличивает погрешность изменений сдвига фазы и в конечном счете приводит к существенным ошибкам измерения дальности с помощью CW метода.

Ошибки измерения дальности из-за переотражений (MPI) излучения подсвета

Когда сцена освещается источником света, излучение каждой точки можно рассматривать как две составляющие – прямую и глобальную.

Прямая составляющая – отражение излучение подсвета от объекта. Глобальная – принятое освещением объекта другими источниками сцены. Рассмотрим точку P сцены, показанную на рисунке 1.

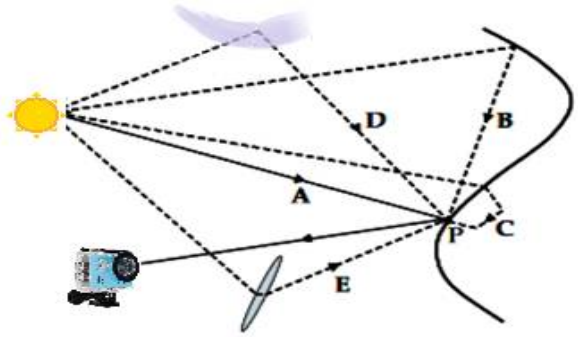


Рис. 1. Сцена освещается источником света

Луч A представляет собой прямое освещение, следовательно, является единственной прямой составляющей излучения, измеряемой камерой. Лучи B, C, D и E сходятся в P от других точек сцены, и вместе они вносят вклад в глобальную составляющую ее излучения, измеряемую камерой.

Эти световые лучи глобального освещения вызваны различными физическими явлениями, распространенными в реальном мире:

- Луч B представляет взаимное отражение света между точками сцены;
- Луч C является результатом рассеяния в среде под поверхностью;
- Луч D обусловлен объемным рассеянием среды-участника в сцене;
- Луч E обусловлен диффузией света через полупрозрачную поверхность.

Необходимо знать способ измерения прямой и глобальной составляющих сцены, так как каждый компонент передает важную информацию. Следовательно, метод, который может измерять только прямую составляющую, может быть использован для расширения широкого спектра методов захвата сцены, которые используются в компьютерном зрении и компьютерной графике.

Глобальный компонент передает сложные оптические взаимодействия между различными объектами и средами в сцене. Именно глобальный компонент делает фотореалистичную визуализацию сложной и интенсивной в вычислительном отношении проблемой. Изме-

рение прямой и глобальной составляющих может обеспечить новые типы манипуляций с изображениями, которые в большей мере соответствуют физическим законам.

Коррекция ошибок

Предлагается для коррекции дальности использовать значения яркости только подсвета ToFсенсора. Для этого следует разделить значение яркости принятого излучения на принятое отраженное излучение подсвета и глобальное (внешнее) освещение. Разделение предлагается выполнить по формулам

$$L_{direct} = Lmax - Lmin, \qquad L_{global} = 2Lmin,$$

где L_{direct} – принятое отраженное излучение подсвета;

L_{global} – внешнее излучение;

$L_{max} = \max(A_0, A_{90}, A_{180}, A_{270})$;

$L_{min} = \min(A_0, A_{90}, A_{180}, A_{270})$.

Малые значения яркости измерений A_0, A_{90}, A_{180}, A_{270} свидетельствуют том, что отраженный сигнал не принят должным образом. Это может быть следствием существенной удаленности объекта, или объект сильно поглощает излучение подсвета. Предлагается найти измерения A0, A90, A180, A270 относящиеся к неустойчивым значениям следующим образом

$$M = \begin{cases} 0, A_{270} - A_{90} < \delta and A_0 - A_{180} < \delta \\ 1, \text{в остальных случаях} \end{cases},$$

где M - маска измерений,

δ - некоторый порог по яркости (предлагается значение $\delta = 10$).

Данное значение позволяет отсечь малые значения яркости, которые скорее всего являются полностью шумовыми. Значения в маске M, равные 1, соответствуют корректным измерениям.

Однако можно попытаться получить информацию о дальности даже для измерений, которые отмечены в маске M нулевыми значениями. Для этого физически надо увеличить время выдержки кадра ToF сенсора, что позволит сенсору накопить больше фотонов излучения и обеспечить в измерении большее значение сигнал/шум. В большинстве случаев повышение выдержки – неприемлемая операция. Тогда можно попытаться собрать фотоны с большей площади (больше одного пикселя сенсора ToF). Это возможно сделать, если поверхность объекта для этих пикселей выглядит гладкой. Предлагается следующий способ интеграции фотонов:

$$A'_{0(j,i)} = \sum_{k=-R}^{k=R} \sum_{m=-R}^{m=R} A_0(j+k, i+m), M(j+k, i+m) \geq \delta.$$

Аналогично должны быть вычислены A'_{90}, A'_{180}, A'_{270}.

Таким образом, можно сделать вывод о том, что предложенные способы повышения точности вычислений дальности показали существенное улучшения результатов измерения дальности камерой ToF.

Библиографический список

1. M. Hansard, S. Lee, O. Choi, R. Horaud. Time-of-Flight Cameras: Principles, Methods and Applications, Springer Brief in Computer Science, 2012.

2. J. Illade-Quinteiro , V. M. Brea , P. López , D. Cabello and G. Doménech-Asensi. Distance Measurement Error in Time-of-Flight Sensors Due to Shot Noise. Sensors 2015, 15(3), 4624-4642; https://doi.org/10.3390/s150304624

3. Lefloch D. et al. (2013) Technical Foundation and Calibration Methods for Time-of-Flight Cameras. In:

4. Grzegorzek M., Theobalt C., Koch R., Kolb A. (eds) Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications. Lecture Notes in Computer Science, vol 8200. Springer, Berlin, Heidelberg

СЕКЦИЯ «ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ»

УДК 004.82; ГРНТИ 20.23.25

СТРУКТУРНОЕ СОПОСТАВЛЕНИЕ RDFS-ОНТОЛОГИЙ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКА SPARQL

В.Н. Дубинин*, А.В. Дубинин*, Ч.-В. Янг, В.В. Вяткин****
** Пензенский государственный университет,*
Россия, Пенза, dubinin.victor@gmail.com
*** Технический университет Лулео,*
Швеция, Лулео, chen-wei.yang@ltu.se

Аннотация. В статье рассматривается метод структурного сопоставления RDFS-онтологий на уровне TBox, позволяющий обнаруживать группы классов с похожими иерархиями. Предлагается реализация данного метода на основе SPARQL-запросов, в которых операции над множествами сведены к манипуляции над числами.
Ключевые слова: сопоставление онтологий, иерархия классов, TBox, RDFS, SPARQL.

STRUCTURAL MATCHING OF RDFS ONTOLOGIES USING SPARQL

V.N. Dubinin*, A.V. Dubinin*, C.-W. Yang, V.V. Vyatkin****
** Penza State University,*
Penza, Russia, dubinin.victor@gmail.com
*** Lulea University of Technology,*
Sweden, Lulea, chen-wei.yang@ltu.se

Abstract. The paper discusses the method of structural matching of RDFS ontologies at the TBox level, which makes it possible to detect groups of classes with similar hierarchies. An implementation of this method is proposed based on SPARQL queries, in which operations on sets are reduced to operations on numbers.
Keywords: ontology matching, class hierarchy, TBox, RDFS, SPARQL.

Прогресс информационных и коммуникационных технологий сделал доступным огромное количество разрозненных источников информации. Это привело к проблеме интеграции семантически гетерогенной информации. Среди основных средств решения этой проблемы – семантические технологии, в числе которых сопоставление онтологий (*ontology mathing*, *ontology alignment*). Сопоставление (иными словами, отображение, согласование, выравнивание) онтологий – это процесс установления соответствий между концептами нескольких онтологий. В последнее время эта область исследований бурно развивается. В 2004 г. стартовала международная инициатива *Ontology Alignment Evaluation Initiative*, направленная на достижение консенсуса в области оценки, сравнения и улучшения различных подходов сопоставления онтологий. В настоящее время существует множество различных обзоров по этой теме (например, [1]).

Для извлечения информации из RDF-хранилищ был предложен ряд языков, среди которых выделяется язык SPARQL, одобренный консорциумом W3C [2]. Не сегодняшний день этот язык является одним из наиболее перспективных для обработки онтологий, в том числе и для их сопоставления. В работе [3] для сопоставления онтологий предлагается ряд расширений языка SPARQL. В работе [4] сходства и различия между сущностями моделируются в виде запросов SPARQL, удовлетворяющих определенным дополнительным свойствам, и предлагаются алгоритмы их вычисления. В статье [5] предлагается эффективный способ получения доступа к хранилищу RDF-данных, даже если отсутствует схема. Для этого дается эффективная модель SPARQL-запроса, возвращающего ответы на основе семантического сходства.

Цель данной работы лежит в практической области – показать возможности языка SPARQL на простом примере структурного сопоставления RDFS-онтологий на уровне TBox. В данном случае обнаруживаются группы классов с похожими иерархиями. Поскольку в

языке SPARQL нет средств для проведения операций над множествами, то предлагается заменить их на манипуляции над числами.

Существуют различные варианты для сравнения классов, основанные на иерархиях. Очевидно, что чем больше глубина иерархии, тем точнее сравнение классов по семантической схожести. Глубина сравнения – это расстояние от данного класса до класса-листа иерархии (включая отношения между классами как элементы пути).

Простейшим случаем является одноуровневый вариант на рис.1,а, когда сравниваются только классы и их свойства. Данный вариант реализован на языке SPARQL и представлен как пример в данной работе. Более сложным является вариант на рис. 1,б. В этом случае, кроме самих свойств, учитываются также значения этих свойств. Развитием и усложнением одноуровневых вариантов являются двухуровневые варианты на рис. 1,в и рис. 1,г. В первом из них учитываются только свойства базовых классов. Второй вариант является усовершенствованием первого. В этом случае, учитываются также классы, которые являются значениями этих свойств.

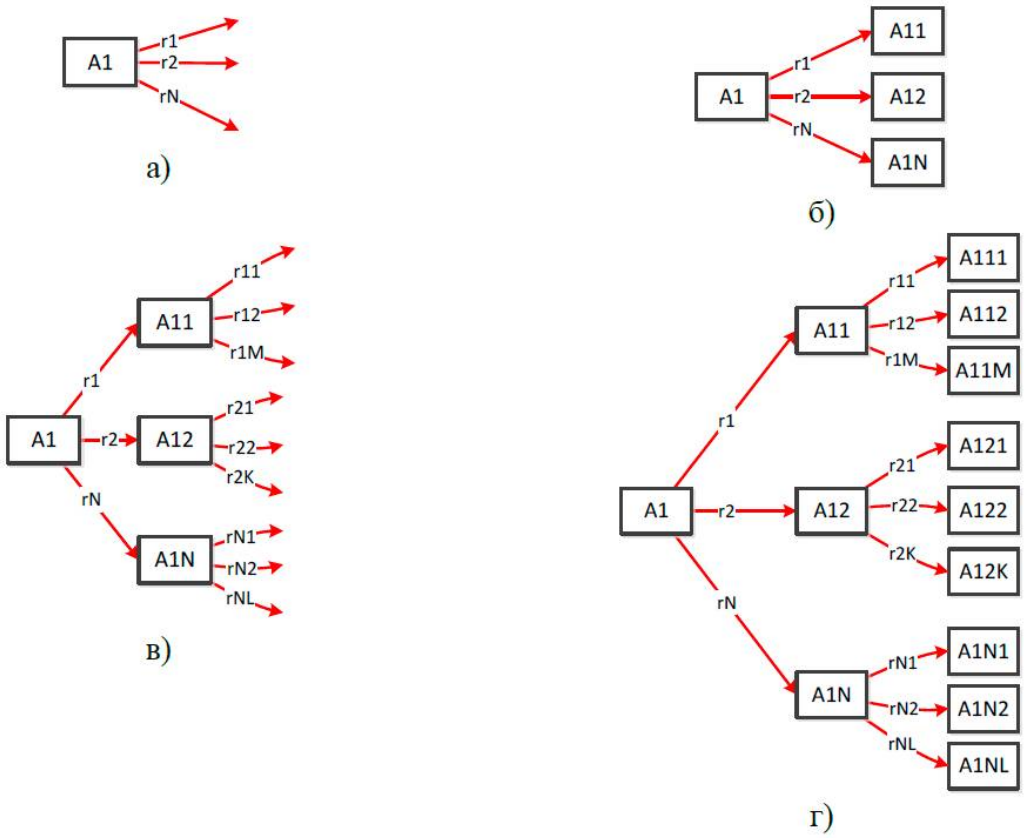


Рис. 1. Варианты классов с различной иерархией

Входная онтология $O^i, i = 1,2$ определяется как тройка: $O^i = (C^i, R^i, Dom^i)$, где $C^i = \{c_1^i, c_2^i, ..., c_n^i\}$ - множество классов; $R^i = \{r_1^i, r_2^i, ..., r_m^i\}$ - множество свойств (объектных свойств и/или свойств по данным); $Dom^i \subseteq R^i \times C^i$ - бинарное отношение, которое связывает свойства с классами-владельцами этих свойств (отношение *domain* в RDFS).

Определим: $R_{c_j^i} = \{x \mid (x, c_j^i) \in Dom^i\}$ - множество свойств класса c_j^i; $N_{c_j^i} = |R_{c_j^i}|$ - число свойств класса c_j^i; $R_{c_i^1, c_j^2} = R_{c_i^1} \cap R_{c_j^2}$ - множество общих свойств классов c_i^1 и c_j^2; $N_{c_i^1, c_j^2} = |R_{c_i^1, c_j^2}|$ - число общих свойств классов c_i^1 и c_j^2

Рассмотрим возможные варианты соответствий между множествами свойств классов двух различных онтологий O^1 и O^2:

M1: $R_{c_i^1} = R_{c_j^2}$. Классы имеют одинаковое множество свойств. Эти классы можно назвать эквивалентными.

M2: $R_{c_i^1} \subset R_{c_j^2}$. Множество свойств класса первой онтологии строго включается в множество свойств класса второй онтологии.

M3: $R_{c_i^1} \subseteq R_{c_j^2}$. Множество свойств класса первой онтологии нестрого включается в множество свойств класса второй онтологии.

M4: $R_{c_j^2} \subset R_{c_i^1}$. Множество свойств класса второй онтологии строго включается в множество свойств класса первой онтологии.

M5: $R_{c_j^2} \subseteq R_{c_i^1}$. Множество свойств класса второй онтологии нестрого включается в множество свойств класса первой онтологии.

M6: $R_{c_i^1} \cap R_{c_j^2} \neq \varnothing \wedge (R_{c_i^1} \cap R_{c_j^2}) \subset R_{c_i^1} \wedge (R_{c_i^1} \cap R_{c_j^2}) \subset R_{c_j^2}$. Множества свойств классов первой и второй онтологий строго частично пересекаются. То есть, оба класса обязательно имеют свои собственные свойства.

M7: $R_{c_i^1} \cap R_{c_j^2} \neq \varnothing \wedge (R_{c_i^1} \cap R_{c_j^2}) \subseteq R_{c_i^1} \wedge (R_{c_i^1} \cap R_{c_j^2}) \subseteq R_{c_j^2}$. Множества свойств классов первой и второй онтологий частично пересекаются (включая случаи эквивалентности и покрытия). Здесь основной принцип: «Есть хоть что-то общее».

M8: $R_{c_i^1} \cap R_{c_j^2} = \varnothing$. Множества свойств классов первой и второй онтологий не пересекаются.

Сведем операции над множествами, приведенными выше, к операциям над числами. Это необходимо, поскольку возможности операций над множествами в языке SPARQL довольно ограничены. Кроме того, это упрощает реализацию на SPARQL, используя предложения GROUP BY.

$R_{c_i^1} = R_{c_j^2}$ эквивалентно $N_{c_i^1, c_j^2} = N_{c_i^1} = N_{c_j^2}$.

$R_{c_i^1} \subset R_{c_j^2}$ эквивалентно $N_{c_i^1} = N_{c_i^1, c_j^2} \wedge N_{c_i^1, c_j^2} < N_{c_j^2}$.

$R_{c_i^1} \subseteq R_{c_j^2}$ эквивалентно $N_{c_i^1} = N_{c_i^1, c_j^2} \wedge N_{c_i^1, c_j^2} \leq N_{c_j^2}$.

$R_{c_i^1} \cap R_{c_j^2} \neq \varnothing \wedge (R_{c_i^1} \cap R_{c_j^2}) \subset R_{c_i^1} \wedge (R_{c_i^1} \cap R_{c_j^2}) \subset R_{c_j^2}$ эквивалентно

$N_{c_i^1, c_j^2} \neq 0 \wedge N_{c_i^1, c_j^2} < N_{c_i^1} \wedge N_{c_i^1, c_j^2} < N_{c_j^2}$.

Пусть заданы две онтологии O1 и O2:

O1=({A,B,C,D,E}, {p1,p2,r1,r2,r3,r4}, {(p1,A), (p1,B), (p2,B), (r1,C), (r1,E), (r2,D), (r2,E), (r3,D), (r4,E)}).

O2=({P,Q,R,S,T}, {q1,q2,q3,r1,r2,r3,r4}, {(r1,P), (r2,Q), (r3,Q), (r1,R), (r2,R), (r4,R), (q1,S), (q2,S), (q1,T), (q3,T)}).

Графическое представление соответствий M1-M6 между классами онтологий приведено на рис. 2.

Для обеспечения независимости кода SPARQL от обрабатываемых онтологий хранилище данных построено на модульной основе. При этом используемые онтологии представлены в виде (независимых) графов G1, G2 и G3. В нашем случае мы используем структуру системы (рис. 3), которая состоит из элементов хранения и элементов обработки. Для удобства мы объединили все графы в один набор данных, названный *into_matching_test1*. В качестве SPARQL-интерпретатора в данном случае использовался сервер *Apache Jena Fuseki* [6].

Для сокращения объема SPARQL-запросов, а также увеличения читабельности, вводим префиксы. Префиксы *m1, m2* and *m3* используются для онтологий, подготовленных в системе *Protégé*. Префикс *g* используется для сокращения имени набора данных.

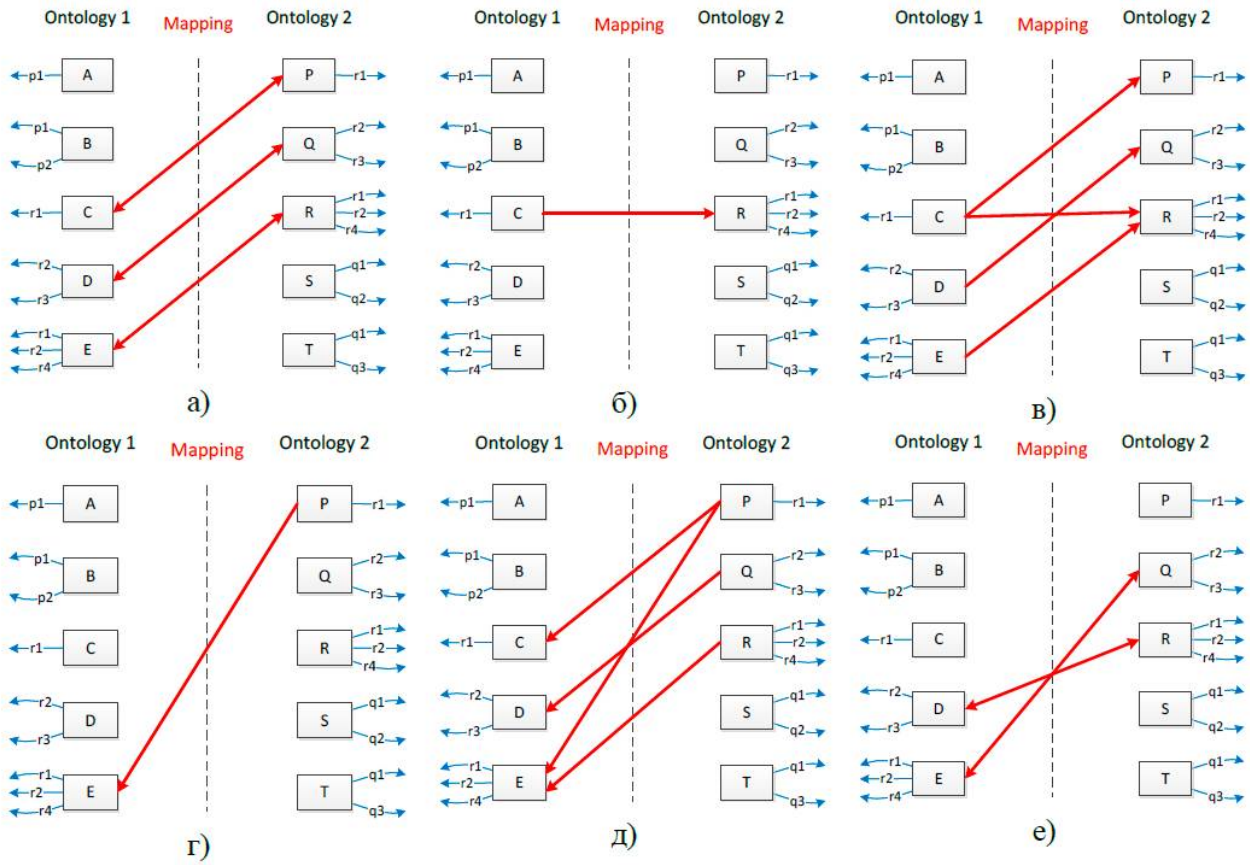


Рис. 2. Соответствия между классами онтологий

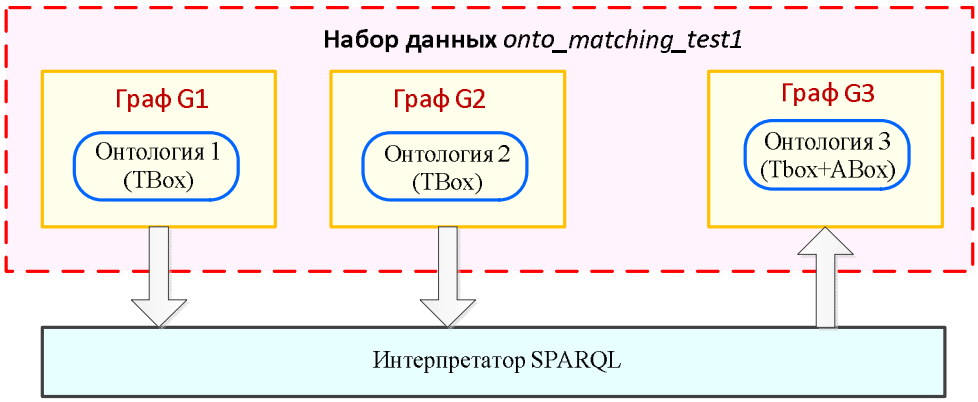


Рис. 3. Структура системы для сопоставления онтологий

SPARQL-запрос, представленный ниже предназначен для вычисления отображения между классами онтологий O1 и O2 в соответствии с M1 (эквивалентность классов) и формировании на его основе выходной онтологии O3.

Особенностью этого запроса SPARQL Update является широкое использование подзапросов и группирований. Первый подзапрос вычисляет количество идентичных свойств для двух классов. Второй подзапрос вычисляет количество свойств первого класса. Третий подзапрос вычисляет количество свойств второго класса.

Чтобы сравнить имена двух свойств (в разных онтологиях), которые представлены как IRI, необходимо избавиться от IRI онтологий, так как они различны по определению. Для этого мы используем функцию SUBSTR (вызывается как SUBSTR (str (? Xxx), 74)). Число 74 определяет здесь длину IRI рассматриваемых онтологий, которые должны быть опущены.

```
PREFIX m1: <http://www.semanticweb.org/victor/ontologies/2019/7/untitled-ontology-42#>
PREFIX m2: <http://www.semanticweb.org/victor/ontologies/2019/7/untitled-ontology-43#>
PREFIX m3: <http://www.semanticweb.org/victor/ontologies/2019/7/untitled-ontology-44#>
PREFIX g: <http://localhost:3030/onto_matching_test1/data/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
INSERT
  {GRAPH g:G3{?k1 m3:mapping ?k2}}
WHERE {
{SELECT ?k1 ?k2 (COUNT (?rez1) AS ?cnt)
WHERE {
  GRAPH g:G1{?r1 rdfs:domain ?k1}
  BIND(SUBSTR(str(?r1),74) AS ?rez1)
  GRAPH g:G2{?r2 rdfs:domain ?k2}
  BIND(SUBSTR(str(?r2),74) AS ?rez2)
  FILTER (?rez1=?rez2) }
GROUP BY ?k1 ?k2 }
{SELECT ?k1 (COUNT (?r1) AS ?cnt1)
WHERE
 {GRAPH g:G1{?r1 rdfs:domain ?k1}}
GROUP BY ?k1 }
{SELECT ?k2 (COUNT (?r2) AS ?cnt2)
WHERE
 {GRAPH g:G2{?r2 rdfs:domain ?k2}}
GROUP BY ?k2 }
  FILTER (?cnt=?cnt1 && ?cnt=?cnt2) }
```

Последняя строка запроса FILTER() определяет тип соответствия, в соответствии с которым должно быть построено отображение. В приведенном SPARQL-запросе - это соответствие M1. При изменении этой строки мы можем строить другие отображения M2-M8. Следует отметить, что переменная ?cnt связана с числом $N_{c_i^1,c_j^2}$ в приведенном выше формальном определении, а переменная ?cnt1 (?cnt2) соотносится с $N_{c_j^1}(N_{c_j^2})$.

В данной работе показана выразительная сила языка запросов SPARQL для структурного сопоставления онтологий. Направлением дальнейших исследований является реализация SPARQL-запросов для структурного сопоставления классов с произвольными иерархиями, а также реализация других стратегий сопоставления.

Библиографический список

1. Shvaiko P., Euzenat J. Ontology matching: state of the art and future challenges // IEEE Transactions on Knowledge and Data Engineering. – 2013. – No. 25 (1). – pp. 158-176.
2. DuCharme B. Learning SPARQL: Querying and Updating with SPARQL 1.1, O'Reilly, 2011, 237 p.
3. Euzenat J., Polleres A., Scharffe F. Processing ontology alignments with SPARQL //Int. Conf. on Complex, Intelligent and Software Intensive Systems, Barcelona, Spain, 2008, pp. 1-6.
4. Petrova A., Sherkhonov E., Grau B.C., Horrocks I. Entity Comparison in RDF Graphs // Lecture Notes in Computer Science. – 2017. – Vol. 10587. – pp. 526-541.
5. Zheng W., Zou L., Peng W., Yan X., Song S., Zhao D. Semantic SPARQL Similarity Search Over RDF Knowledge Graphs // Proc. VLDB Endowment, Vol. 9, No. 11, 2016, pp. 840-851.
6. Apache Jena Fuseki [Электронный ресурс]. – Режим доступа: https://jena.apache.org/documentation/fuseki2/. – Дата доступа: 01.08.2019.

УДК 004.89; ГРНТИ 50.07.03

ОНТОЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ СЕТЕЙ ПЕТРИ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКА SPARQL

В.Н. Дубинин*, А.В. Дубинин*, Л.П. Климкина**
** Пензенский государственный университет,*
Россия, Пенза, dubinin.victor@gmail.com
*** Пензенский государственный аграрный университет,*
Россия, Пенза, ludmila.klimkina@gmail.com

Аннотация. В работе предлагается онтологический подход к моделированию сетей Петри с использованием языка SPARQL. Рассматривается онтология сетей Петри, SPARQL-запросы, моделирующие динамику работы, а также вопросы построения клиент-серверных систем для интерпретации сетевых моделей.
Ключевые слова: онтология, сети Петри, моделирование, динамика, SPARQL.

ONTOLOGICAL MODELLING OF PETRI NETS USING SPARQL LANGUAGE

V.N. Dubinin*, A.V. Dubinin*, L.P. Klimkina**
** Penza State University,*
Penza, Russia, dubinin.victor@gmail.com
*** Penza State Agricultural University,*
Penza, Russia, ludmila.klimkina@gmail.com

Abstract. The paper proposes an ontological approach to modeling Petri nets using the SPARQL language. An ontology of Petri nets, SPARQL queries for modeling the dynamics of functioning, as well as issues of building client-server systems for interpreting the net models are considered.
Keywords: ontology, Petri nets, modelling, dynamics, SPARQL.

В настоящее время сети Петри (СП) являются хорошо апробированным и эффективным средством моделирования, анализа и синтеза распределенных и параллельных дискретных событийных систем [1]. Было предложено множество различных модификаций и расширений СП, в частности, сетевой формализм (СФ) [2]. СФ обладает свойствами временных, самомодифицирующихся, приоритетных, числовых сетей Петри. В него включены ингибиторные дуги, прерываемые переходы, ограниченные позиции, метки с атрибутами и без атрибутов, переменные и процедуры переходов. СФ имеет следующие оригинальные расширения: процедуру структурной модификации сетевой модели, рабочие подочереди, переходы с ИЛИ-И логикой.

Кроме моделирования, анализа и синтеза, СП и производные от них модели используются для реализации на различных программно-аппаратных платформах. Например, функ-

ционально-блочная реализация СП для архитектуры IEC 61499 рассмотрена в работе [3]. Причины использования СП для их реализации следующие: 1) удобство, наглядность и компактность в описании параллельных процессов со сложными видами взаимодействий; 2) формальность модели, что исключает двусмысленности в описании и позволяет при необходимости проводить верификацию и оценку производительности системы.

Активно развиваемая концепция семантического Web предполагает особую архитектуру распределенной системы, во многом основывающуюся на онтологиях [4]. Для описания онтологий наиболее часто используются языки RDFS и OWL. В любом случае онтологию на нижнем уровне можно представить с использованием RDF. Основным понятием RDF является триплет, а в целом RDF-описание представляет собой граф (или RDF-граф) [5]. Для извлечения информации из RDF-хранилищ был предложен ряд языков, среди которых выделяется язык SPARQL, одобренный консорциумом W3C [6]. В 2013 г. была принята текущая версия SPARQL 1.1, одним из важнейших расширений которой является включение в состав языка средств манипулирования RDF-данными (SPARQL 1.1 Update).

На данный момент существует небольшое число работ, в которых предлагается реализация СП с использованием онтологий и семантических технологий. Так, например, в работе [7] разрабатывается онтология СП для создания инфраструктуры СП в семантическом Web, которая, в свою очередь, должна обеспечить интероперабельность и совместное использование СП в семантической паутине. В работе [8] представлен “движок” OPENET для выполнения высокоуровневых СП. OPENET основан на разработанной онтологии высокоуровневых СП, которая семантически и декларативно описывает как статическую структуру, так и динамическое поведение. Этот «движок» реализован на основе формального аппарата F-Logic с использованием ризонера FLORA.

В отличие от вышеупомянутых работ, в данной работе предлагается онтологический подход к моделированию СП с использованием языка SPARQL. Использование этого языка имеет ряд преимуществ: 1) позволяет напрямую моделировать динамику СП, представленных в виде онтологий. Данная возможность обеспечивается подязыком SPARQL Update для манипулирования данными; 2) язык SPARQL де-факто является стандартным языком запросов семантического Web, что способствует относительно легкой интеграции SPARQL-реализации СП с другими приложениями семантического Web; 3) язык SPAPQL хорошо приспособлен для трансформации онтологий, что позволяет преобразовывать онтологические описания СП на стадии проектирования, поддерживая, таким образом, подход на основе управления онтологиями. Для выполнения и тестирования SPARQL-запросов в данной работе использовался сервер *Apache Jena Fuseki* [9].

Рассмотрим моделирование и реализацию СП с использованием онтологий. Особенностью рассматриваемых СП являются наличие числовых и ингибиторных дуг, а также приоритетов переходов. Таким образом, данные СП являются подклассом СФ [2] и определяются следующим кортежем:

$$(P, T, X, Y, Z, W_X, W_Y, W_Z, Q, A, G, m_0),$$

где P – конечное множество позиций;

T – конечное множество переходов;

$X \subseteq P \times T$ – множество входных дуг переходов с порогом-минимумом (дуги с проверкой на “больше”);

$Y \subseteq T \times P$ – множество выходных дуг переходов;

$Z \subseteq P \times T$ – множество входных дуг переходов с порогом-максимумом (дуги с проверкой на “меньше”);

W_X: $X \rightarrow N^+ \times N_0$ – функция весов дуг типа X;

W_Y: $Y \rightarrow N^+$ – функция весов дуг типа Y;

W_Z: $Z \rightarrow N^+$ – функция весов дуг типа Z;

Q: $T \rightarrow N_0$ – функция приоритетов переходов (чем меньше значение, тем выше приоритет);

m_0 – начальная маркировка. Первый компонент веса дуги типа X указывает минимальный порог для срабатывания перехода, а второй – число удаляемых меток из инцидентной дуге позиции.

Разработанная онтология уровня *TBox* для данного класса сетевых моделей графичесик представлена на рисунке 1.

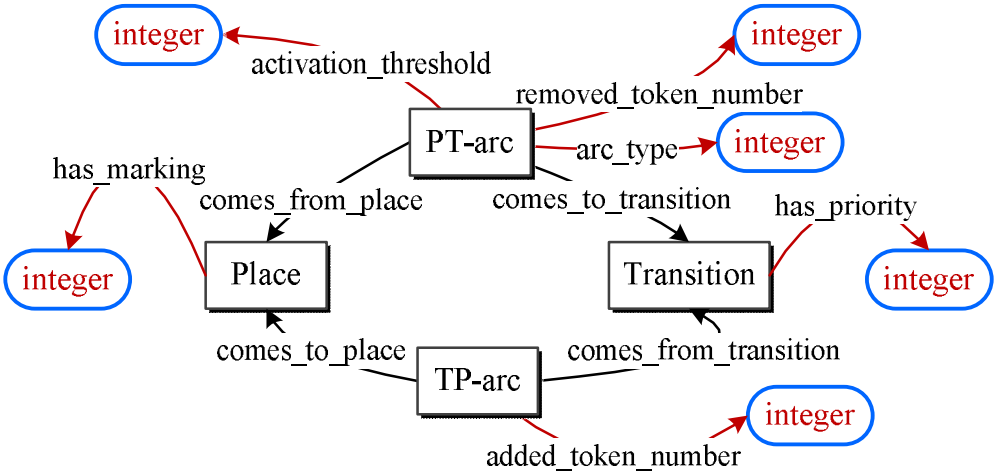


Рис. 1. Онтология TBox сетей Петри

Она включает: четыре класса - *Place* (Позиция), *Transition* (Переход), *PT-arc* и *TP-arc* (входная и выходная дуга перехода, соответственно); четыре объектных свойства – *comes_from_place, comes_from_transition, comes_to_place* и *comes_to_transition*, соответствующие дугам сетевой модели и показывающие источник и приемник дуги; шесть свойств по данным – *has_marking* (маркировка позиции), *arc_type* (тип дуги), *activation_threshold* (порог активизации входной дуги перехода), *removed_token_number* (число удаляемых из позиции меток при срабатывании перехода), *added_token_number* (число добавляемых в позицию меток при срабатывании перехода) и *has_priority* (приоритет перехода).

На рис. 2 в качестве примера приведена простая СП, а на рис. 3 - онтология уровня ABox для данной СП, сгенерированная в плагине *Ontograf* системы Protege.

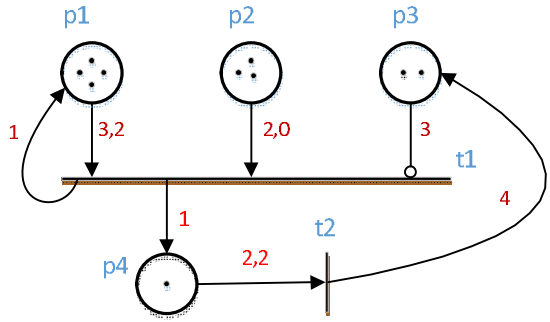


Рис. 2. Пример сети Петри

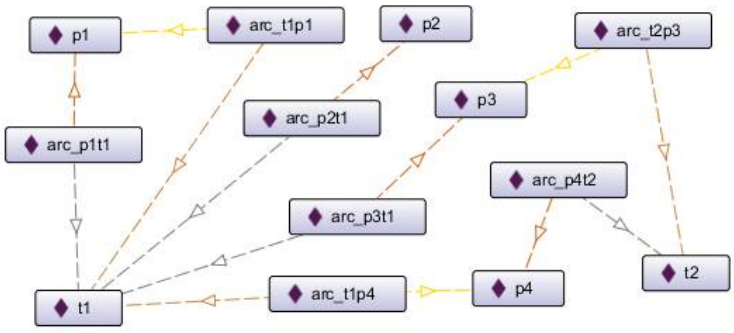


Рис. 3. Онтология ABox сети Петри, приведенной на рисунке 2

С использованием языка SPARQL Update можно промоделировать только одно срабатывание перехода. Однако даже эта задача является нетривиальной по ряду причин: 1) каждый конкретный переход имеет свою конфигурацию (т.е. множества входных и выходных дуг); 2) одна и та же позиция может быть для перехода одновременно входной и выходной. Для разрешения конфликта, когда в позицию одновременно должны добавляться и удаляться метки, класс входо-выходных позиций рассматривается отдельно. Соответственно, новая маркировка этих позиций вычисляется по отдельной формуле.

SPARQL-запрос, моделирующий срабатывание перехода СП, имеет следующую структуру: 1) фрагмент кода для удаления триплетов маркировки; 2) фрагмент кода для добавления триплетов маркировки; 3) подзапрос для определения активного перехода; 4) фрагмент кода для подсчета новой маркировки входной или входо-выходной позиции; 5) фрагмент кода для подсчета новой маркировки выходной позиции. Первые два фрагмента приведены ниже:

```
DELETE
{?p1 m:has_marking ?n1.    # Удалить прежнюю маркировку входных и входо-выходных позиций
?p2 m:has_marking ?n2.}    # Удалить прежнюю маркировку выходных позиций
INSERT
{?p1 m:has_marking ?n1new.  # Включить новую маркировку входных и входо-выходных позиций
?p2 m:has_marking ?n2new.}  # Включить новую маркировку выходных позиций
```

В запросах префикс *m* представляет входную онтологию СП типа TBox+ABox. Под активным переходом понимается разрешенный переход с максимальным приоритетом, который будет срабатывать. Переход разрешен в том случае, когда число входных дуг перехода равно числу разрешенных входных дуг перехода. При вычислении подзапроса используется группирование (GROUP BY), фильтрация (FILTER), упорядочение (ORDER BY) и ограничение числа результатов (LIMIT). В самом подзапросе используются два других подзапроса. Подзапрос для определения активного перехода представлен ниже.

```
{SELECT ?t
WHERE
{?t rdf:type m:Transition.
?t m:has_priority ?pr.
{SELECT ?t (COUNT(?a) AS ?na)   # Подсчет числа входных дуг перехода
WHERE
{?a m:comes_to_transition ?t}
GROUP BY ?t}
{SELECT ?t (COUNT(?a) AS ?ne)   # Подсчет числа разрешенных входных дуг перехода
WHERE
{?a m:comes_to_transition ?t. ?a m:comes_from_place ?p.
?p m:has_marking ?n. ?a m:activation_threshold ?k. ?a m:arc_type ?s.
FILTER (?s='n' && ?n>=?k || ?s='i' && ?n<=?k)}
GROUP BY ?t }
```

```
FILTER (?na=?ne)}  # Отбор: У разрешенного перехода все дуги должны быть разрешенными
ORDER BY ?pr       # Упорядочить разрешенные переходы по убыванию приоритетов
LIMIT 1}
```

Ниже представлен фрагмент кода, определяющий, является ли входная позиция ?p1 также и выходной, и вычисляющий новую маркировку позиции ?p1. В данном фрагменте переменная ?k1 определяет число удаляемых меток.

```
OPTIONAL   # Определяем, является ли входная позиция ?p1 также и выходной
{?a2 rdf:type m:TP-arc.
?a2 m:comes_from_transition ?t.
?a2 m:comes_to_place ?p1.
?a2 m:added_token_number ?k2a.}
BIND ((IF (bound(?k2a),?n1-?k1+?k2a, ?n1-?k1)) AS ?n1new)  # Подсчет новой маркировки
```

Фрагмент кода для подсчета новой маркировки чисто выходной позиции приведен ниже.

```
OPTIONAL    # Определяем выходные позиции ?p2 перехода ?t
{?a3 rdf:type m:TP-arc.
?a3 m:comes_from_transition ?t.
?a3 m:comes_to_place ?p2.
?a3 m:added_token_number ?k2b.
?p2 m:has_marking ?n2.}
MINUS   # Вычитаем множество тех выходных позиций, которые также являются и входными
{?a4 rdf:type m:PT-arc.
?a4 m:comes_to_transition ?t.
?a4 m:comes_from_place ?p2.}
BIND (?n2+?k2b AS ?n2new)   # Подсчет новой маркировки выходной позиции ?p2
```

Вспомогательный SPARQL-запрос для вывода текущей маркировки СП, используемый в ходе тестирования, может быть следующим.

```
SELECT ?p ?n
WHERE
{?p rdf:type m:Place.
?p m:has_marking ?n.}
```

Выполнение данного *query*-запроса должно чередоваться с выполнением *update*-запроса на срабатывание перехода СП. При моделировании СП, приведенной на рис. 2, была получена единственная трасса выполнения СП: (4,3,2,1)-t1-(3,3,2,2)-t1-(2,3,2,3)-t2-(2,3,6,1).

Правило (вернее, серия правил) в SPARQL-системе выполняется однократно. Поскольку все транзакции SPARQL являются линейными (так как операторы управления выполнением транзакций отсутствуют), то в лучшем случае можно выполнить последовательность из серий правил перезаписи графов. Для организации циклических стратегий управления выполнением правил необходима разработка внешних программ (клиентов). Существует два основных подхода к организации взаимодействия подобного клиента со SPARQL-сервером: 1) использование протокола HTTP; 2) использование API сервера или специальных классов.

Для интерпретации СП на основе разработанных SPARQL-запросов было разработано несколько клиентских программ, реализованных на основе разных подходов и с использованием разных языков программирования (для сравнительного анализа). В первом случае клиентская программа была реализована на языке *C#* с использованием HTTP-взаимодействий (метод POST). Функции клиента следующие: чтение онтологии СП из файла, визуальное отображение СП с текущей маркировкой в виде двудольного графа на экране (с использованием компонента *MindFusion Diagramming*), посылка SPARQL-запроса на срабатывание пе-

рехода СП на сервер, посылка SPARQL-запроса на выборку новой маркировки, сброс сети к исходному состоянию. Клиент-серверная система в целом работает интерактивно, в пошаговом режиме – нажатие кнопки в ЧМИ приводит к срабатыванию одного перехода. Второй клиент был разработан на языке *JavaScript* с использованием библиотеки *jQuery*. В отличие от первого клиента, во втором клиенте нет графического отображения СП, выводится лишь маркировка сети в виде таблицы. Однако при этом реализовано не только пошаговое выполнение, но и режим выполнения с заданным числом срабатываний переходов, а также режим выполнения до фиксированной точки (*fixpoint*). Для многих языков программирования (например, *C#*, *Python*) существуют библиотеки классов для работы с RDF. Например, подключение в серверу *Apache Jena Fuseki* может быть реализовано на C# как

```
FusekiConnector fuseki = new FusekiConnector("http://localhost:3030/dataset/pn");
```

Направлением дальнейших исследований является разработка метода онтологического моделирования высокоровневых сетей Петри с использованием языка SPARQL.

Библиографический список

1. Питерсон, Дж. Теория сетей Петри и моделирование систем. – М. : Мир, 1984. – 263 с.

2. Дубинин В.Н. Разработка архитектуры, методов и средств проектирования периферийных систем управления ЭВМ. – Дисс. на соиск. уч. степ. канд. техн. наук. – Пенза, 1988. – 308 с. [Электронный ресурс]. – Режим доступа: https://drive.google.com/file/d/0B1_0eFPTb3VXaS1DWElWWllGVjQ/view. – Дата доступа: 01.02.2020.

3. Дубинин В.Н., Сенокосов И.В., Войнов А.С., Вяткин В.В. Функционально-блочная реализация селектирующих сетей Петри // Международный научно-технический форум «Современные технологии в науке и образовании» (СТНО-2018): Сб. трудов. Том 6, Рязань, С. 124-129.

4. Semantic Web. W3C Consortium. [Электронный ресурс]. – Режим доступа:: https://www.w3.org/standards/semanticweb/. – Дата доступа: 01.02.2020.

5. Allemang D., Hendler J. Semantic Web for the Working Ontologist. Modeling in RDF, RDFS and OWL. – Morgan Kaufmann Publishers, 2008. – 349 p.

6. DuCharme B. Learning SPARQL: Querying and Updating with SPARQL 1.1. – O'Reilly, 2011. – 237 p.

7. Gašević, D. Interoperable Petri Net Models via Ontology / D. Gašević, V. Devedžić // International Journal of Web Engineering and Technology. – Vol. 3, N 4, 2007. – P. 374-396.

8. Vidal J.C., Lama M., Bugarín A. OPENET: Ontology-based engine for high-level Petri nets // Expert Systems with Applications. – 2010. – N 37. – P. 6493–6509.

9. Apache Jena Fuseki [Электронный ресурс]. – Режим доступа: https://jena.apache.org/documentation/fuseki2/. – Дата доступа: 01.08.2019.

УДК: 519.24

ПОДХОД К ИДЕНТИФИКАЦИИ ДЛИНЫ ЦИКЛА В СИМВОЛЬНЫХ ПОСЛЕДОВАТЕЛЬНОСТЯХ С ШУМОМ, ОСНОВАННЫЙ НА ЭНТРОПИИ СЛОВ[1]

М.В. Ульянов

Институт проблем управления им. В.А. Трапезникова РАН,
Российская Федерация, Москва, muljanov@mail.ru

Аннотация. В работе рассматривается подход к определению длины периода в периодических символьных последовательностях с шумом. Предлагается использовать для анализа символьной последовательности аппарат энтропии слов. Подход для оценки длины периода основан на особенностях частотной встречаемости слов в окне фиксированной длины, пробегающем по исследуемому слову со сдвигом один.

Ключевые слова: периодические символьные последовательности, шум замены, шум вставки, шум удаления, энтропия слов, длина периода.

[1] Работа выполнена при поддержке гранта РФФИ 19-07-00151.

APPROACH TO IDENTIFICATION OF PERIOD LENGTH IN CHARACTER SEQUENCES WITH NOISE, BASED ON ENTROPY OF WORDS

M.V. Ulyanov
V.A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences
Russia, Moscow, muljanov@mail.ru

Abstract. In the paper, an approach to determining the length of a period in periodic character sequences with noise is considered. It is proposed to use the entropy of words for the analysis of a character sequence. The approach for estimating the period length is based on the features of the frequency of the occurrence of different words in a window of fixed length that moves with a shift of one along the word under consideration.
Keywords: periodic character sequence, almost periodic sequence, noise of replacement, noise of insertion, noise of deletion, entropy of words, period length

Символьные последовательности могут быть эффективно применены для решения задач качественного анализа в прикладных областях, связанных с анализом больших данных. В целях качественного анализа данные целесообразно кодировать символами конечного алфавита, что позволяет отбросить лишние детали, не несущие полезной информации [1]. Заметим, что значительное число исследуемых в настоящее время процессов обладает циклическим характером, и достаточно часто исходные данные подвержены воздействию случайных искажений, вызванных внешними факторами. Эта особенность в аспекте символьного кодирования приводит к необходимости использования модели символьной последовательности с зашумленными циклами. Отметим, что интерес к разработке методов исследования периодичности во временных рядах, искаженных шумом, каждым годом только растет, в первую очередь, из-за широкого перечня задач, которые могут быть решены или позволят улучшить существующие решения. В этот перечень входит прогнозирование погоды, прогнозирование котировок на фондовых биржах, анализ различных особенностей клинических заболеваний, анализ кардиограмм, исследования в биоинформатике, например, поиск скрытых периодов в ДНК, а также другие задачи, связанные с анализом временных рядов [2].

Сложность в нахождении периода возникает при добавлении различных видов шумов в циклическую последовательность, таких как вставка, замена и удаление символов, которые исключают возможность использования классических алгоритмов поиска периода. Отметим, что шумы вставки и удаления искажают не только последовательность символов, составляющую период, но и его длину, что вносит дополнительные сложности в анализ. Поэтому корректно говорить о задаче поиска шаблона цикла и о задаче поиска периодичности. Задача поиска длины периода требует уточнения понятия длины периода, поскольку длина повторяющихся фрагментов, искаженных шумом, будет различаться. При шуме замены мы имеем некоторый шаблон с длиной равной периоду, каждая реализация которого может содержать искажения некоторых символов с некоторыми вероятностями. Для решения задачи поиска периодичности в зашумленных данных в настоящее время существует ряд алгоритмов и методов, а именно: алгоритм WARP [3], алгоритм CONV [4], алгоритм STNR [5], метод, предложенный коллективом авторов под руководством Е.В. Короткова [6]. Проведенный нами анализ данных алгоритмов [7] показал, что они различно реагируют на шумы разных типов. В этом аспекте интерес представляет разработка подхода, инвариантного к типу шума. Предлагается использовать аппарат энтропии слов, включающий в себя подсчет частотной встречаемости подслов.

Функция энтропии конечных слов

Рассмотрим вначале классическую дискретную вероятностную модель с конечным носителем $M = \langle \Omega, P(\cdot) \rangle$, в которой вероятностное пространство Ω дискретно и конечно —

$\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$, $P(\cdot): P(\omega_i) = p_i$, $P(\Omega) = 1$. Тогда, по определению [8] энтропия такого распределения H_P:

$$H_P \overset{def}{=\!=} -\sum_{i=1}^{k} p_i \log p_i . \tag{1}$$

При этом максимум энтропии достигается для равномерного распределения [8] $P(\omega_i) = 1/|\Omega| = 1/k$. Для нормировки максимального значения энтропии (1) в единицу необходимо выбрать основание логарифма, равное мощности вероятностного пространства, тогда:

$$H_p = -\sum_{i=1}^{k} p_i \log_{|\Omega|} p_i = -\sum_{i=1}^{k} \frac{1}{k} \log_k \frac{1}{k} = 1, \tag{2}$$

будем называть такую энтропию нормированной. При распределении, отличном от равномерного, значение $0 \le H_P < 1$, а для вырожденного распределения $\exists \omega_i : P(\omega_i) = 1,\ \forall \omega_j : j \ne i\ \ P(\omega_j) = 0$ энтропия равна нулю.

На этой основе вводится энтропия конечных слов. Введем в рассмотрение оператор сдвига один $SH1(w,k)$, действующий на слово w окном ширины k. Определенный при $|w| \ge k$ оператор порождает мультимножество подслов длины k, с мощностью $|w| - k + 1$, выполняя сдвиг на единицу окна ширины k по слову w, начиная с крайней левой позиции слова w.

Построение функции энтропии слов $H(w,k)$ для исследуемого слова w длины n основано на вычислении энтропии слова w по подсловам фиксированной длины k. Применим к исследуемому слову w оператор сдвига один с окном ширины k и получим порожденное оператором $SH1(w,k)$ мультимножество $\widetilde{V} = SH1(w,k) = \left\{ v_i^{(c_i)} \mid i = \overline{1,l} \right\}$, здесь значение l — число уникальных подслов. На его основе построим вероятностную модель, отражающую частотную встречаемость слов длины k в слове w:

$$P_k(\cdot): \begin{cases} p(\omega_i) = \dfrac{c_i}{m}, & \omega_i \in \widetilde{V} \\ p(\omega_i) = 0, & \omega_i \notin \widetilde{V} \end{cases}, \ m = \sum_{i=1}^{l} c_i .$$

Для этой модели определим нормированную энтропию, используя (2)

$$H_{P_k} = -\sum_{i=1}^{|\Sigma|^k} p_i \log_{|\Sigma|^k} p_i = -\sum_{i=1}^{m} \left(\frac{c_i}{m} \right) \log_{|\Sigma|^k} \left(\frac{c_i}{m} \right). \tag{3}$$

На основе энтропии (3) вводится функция энтропии слов $H(w,k) = H_{P_k}$, аргументом которой является ширина окна (длина подслова) k, с областью определения: $1 \le k \le n$. Заметим, что не зависимо от алфавита Σ и конкретного слова w значение $H(w,n) = 0$, поскольку в окне ширины n мы наблюдаем всего одно слово с вероятностью 1.

Особенности частотной встречаемости слов в символьных последовательностях

Исследуем поведение функции энтропии для случайных и периодических слов, обратив внимание на то, что собственно значение $H(w,k)$ определяется значениями счетчиков c_i уникальных подслов, порождаемых оператором сдвига один — $SH1(w,k)=\left\{v_i^{(c_i)} \mid i=\overline{1,l}\right\}$. Оценим характер поведения счетчиков и функции энтропии для случайных, периодических и периодических зашумленных длинных слов в бинарном алфавите $\Sigma=\{0,1\}$.

1. *Случайное слово.* Пусть слово w получено равномерным генератором псевдослучайных чисел. В этом случае начальное значение $H(w,1)\approx 1$, поскольку символы бинарного алфавита имеют слабо отличающуюся частотную встречаемость в словах большой длины. Значение $H(w,2)\approx 1$, поскольку возможные подслова длины два в исследуемом слове также имеют близкую частотную встречаемость, и.т.д. Тем самым близкие к единице значения $H(w,k)$ в области малых значений аргумента k характеризуют исследуемое слово, как слово, обладающее достаточно богатым разнообразием в области подслов малой длины. Иными словами, в соответствии с (3) это означает, что все подслова бинарного алфавита длины k (для малых значений k) встречаются при сдвиге один (т.е. $m=2^k$) и имеют близкие значения счётчиков c_i.

Однако функция $H(w,k)$ не может долго сохранять близкие к единице значения. Определим пороговое значение $\hat{k}$ для бинарного алфавита, при котором теоретически функция энтропии слов еще может быть очень близка к единице. Для бинарного алфавита в окне ширины k может наблюдаться максимально 2^k различных подслов. Всего в слове длины n мы имеем $n-k+1$ позиций окна сдвига один. Тогда максимально возможная длина подслова при котором еще можно наблюдать полное разнообразие, определяется из уравнения $2^{\hat{k}}=n-\hat{k}+1$, что с учетом целочисленности $\hat{k}$ приводит к уравнению $\hat{k}=\left\lfloor \log_2\left(n-\hat{k}+1\right)\right\rfloor$. В предположении, что $\hat{k}<<n$, пороговое значение $\hat{k}\approx\lfloor \log_2 n\rfloor$. При $n=2^{20}$ пороговое значение $\hat{k}=20$. В этом случае значение $H(w,20)$ будет близко к единице, если в каждой позиции окна сдвига один мы будем наблюдать различные подслова практически без повторений, т.е. значение счетчиков c_i для большинства подслов равно 1. Это характеристика случайной последовательности. Заметим, что в окне ширины $\hat{k}+1$ максимально возможное разнообразие слов в два раза больше полного разнообразия в окне ширины $\hat{k}$. Поэтому мы ожидаем уменьшение значения функции $H(w,k)$ при $k=\hat{k}+1$ даже для псевдослучайной последовательности символов в исследуемом слове.

Если функция энтропии слов начинает убывать уже при ширине окна $k<\hat{k}$, то это означает, что, начиная с данного значения k, разнообразие подслов уменьшилось, и исходное слово обладает определенными особенностями, например, периодичностью.

2. *Периодическое слово*. Пусть слово w является периодическим с периодом l, который многократно повторяется до достижения длины n. Допустим, что подслово длины l, задающее период, также получено равномерным генератором псевдослучайных чисел. Рассуждения, аналогичные приведенным выше, приводят к тому, что максимально возможное значение ширины окна, при котором $H(w,k)$ еще может быть равно или слабо отличаться от единицы, задается формулой $\hat{k}\approx\lfloor \log_2 l\rfloor$. Если, например $l=64$, то $\hat{k}=6$. Таким образом, периодические слова характеризуются тем, что убывание функции энтропии слов начинается существенно раньше, чем у слов, символы которых равномерно случайны. Более того, по

точке начала убывания значений $H(w,k)$, в предположении о случайном характере подслова, образующего период, можно делать предположения об оценке длины этого периода. Заметим также, что в этом случае мы можем наблюдать максимально только l различных подслов в окне сдвига один, что также позволяет оценить длину периода.

3. *Периодическое слово с шумом*. Оценим характер поведения счетчиков c_i для периодического слова, в которое бы внесен небольшой шум замены, удаления или вставки. Пусть период исходной последовательности равен l (образующее слово длины l равномерно случайно) и он многократно повторяется до достижения длины n. После чего в слово w вносится случайный шум замены, удаления и вставки. Отметим, что для окна $k >> \lfloor \log_2 n \rfloor$ разнообразие различных подслов существенно больше, чем возможное число позиций. Рассмотрим возможное поведение счетчиков. Заметим, что внесение единичного шума замены, вставки или удаления приведет, скорее всего, к появлению k новых подслов длины k в операторе сдвига один. И, что очень важно, это не связано с типом вносимого шума. Если упорядочить значения полученных счетчиков по убыванию, то, скорее всего, мы будем наблюдать l «больших» значений c_i со значениями, несколько меньшими, чем $\lfloor n/l \rfloor$ и ряд «малых» значений c_i, определяемых внесенным шумом. Таким образом, если установить порог для учитываемых значений счетчиков, то мы восстанавливаем длину периода зашумленной последовательности при небольшом уровне шума.

Таким образом, предлагаемый подход основан на анализе гистограммы частотной встречаемости подслов в операторе сдвига один при ширине окна $k >> \lfloor \log_2 n \rfloor$. Очевидный интерес представляет изучение порога распознаваемости, т.е. уровня шума, при котором данный подход еще может идентифицировать период. Отличительной особенностью предлагаемого подхода является его инвариантность к типу шума.

Библиографический список

1. Galina Zhukova, Yury Smetanin and Mikhail Uljanov Informative Symbolic Representations as a Way to Qualitatively Analyze Time Series // PROCEEDINGS 2019 International Conference on Engineering Technologies and Computer Science: Innovation & Application pp. 43-47

2. Chanda A.K. et al. A new framework for mining weighted periodic patterns in time series databases // Expert Systems with Applications. – 2017. – Т. 79. – pp. 207-224.

3. Elfeky M.G., Aref W.G., Elmagarmid A.K. WARP: time warping for periodicity detection // Data Mining, Fifth IEEE International Conference on. – IEEE, 2005. –. pp 8.

4. Rasheed F., Alhajj R. STNR: A suffix tree based noise resilient algorithm for periodicity detection in time series databases // Applied Intelligence. – 2010. – Т. 32. – №. 3. – pp. 267-278.

5. Ukkonen E. On-line construction of suffix trees // Algorithmica. – 1995. – Т. 14. – №. 3. – pp. 249-260.

6. Korotkov E.V., Korotkova M.A. Developing new mathematical method for search of the time series periodicity with deletions and insertions // Journal of Physics: Conference Series. – IOP Publishing, 2017. – Т. 788. – №. 1. – pp. 12-19.

7. Жукова Г.Н., Сметанин Ю.Г., Ульянов М.В. Сравнение основных алгоритмов поиска циклов в символьных последовательностях при наличии искажений // Современные информационные технологии и ИТ-образование. 2019. Т. 15, № 4. С. 880-891.

8. Хинчин А.Я. Понятие энтропии в теории вероятностей // Успехи математических наук. — 1953. вып. 3 (55). С. 3 – 20.

УДК 004.42; ГРНТИ 50.05.19

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ОРГАНИЗАЦИИ СТРУКТУРИРОВАННОГО ХРАНЕНИЯ ОБЪЕКТОВ КОНФИГУРАЦИИ

В.В. Белов, О.В. Крылова

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, ole4kakrylova@yandex.ru

Аннотация. В работе рассматривается проектирование и разработка программного обеспечения для повторного использования объектов конфигурации и их хранения. Описываются требования к программной системе и реализованные функции.

Ключевые слова: платформа «1С», объекты конфигурации, повторное использование кода, системы хранения разработанных файлов.

STRUCTURED STORING CONFIGURATION OBJECTS SOFTWARE DESIGN AND DEVELOPMENT

V.V. Belov, O.V. Krylova

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, ole4kakrylova@yandex.ru

The summary. The paper discusses the problems of reuse of configuration objects and their storage facilities. The requirements for the software system and implemented functions are described.

Keywords: platform «1C», configuration objects, code reuse, developed files storage systems.

В фирмах, занимающихся разработкой на платформе «1С», значительная часть заказов поступает на адаптацию типовых решений (таких, как «1С:Бухгалтерия», «1С:Зарплата и управление персоналом» и т.д.) под экономические потребности и бизнес-процессы заказчика (определенной организации или предприятия).

Одной из особенностей доработки типовых конфигураций является схожесть требуемых для различных организаций, работающих в определенной сфере, объектов конфигурации (отчетов, обработок и документов). Аналогичные отчеты, обработки и документы рационально разрабатывать не заново, а используя ранее разработанные решения для данной сферы деятельности [1].

К примеру, если фирма заказчика специализируется в сфере образовательных услуг, и приобрела конфигурацию «1С:Бухгалтерия» с целью автоматизации ведения финансового учета, то в ходе и после ее внедрения, скорее всего, потребуется внести корректировки в печатные формы документов и отчетности, характерные для данного рода деятельности.

При дальнейших заказах компаний сферы образования аналогичные отчеты, обработки и документы рационально разрабатывать не заново, а используя ранее разработанные решения для данной сферы деятельности. Повторное использование кода сокращает время разработки, тем самым повышая эффективность работы разработчиков и снижая оплату для заказчика.

Эффективное хранение разработанных объектов конфигурации предоставляет организации возможность облегчить работу программистов, способствует повышению эффективности их работы, позволяет выполнить заказанные доработки в установленные договором сроки и ввести доработки в эксплуатацию в кратчайшие сроки. Кроме того, удобная система хранения позволит специалисту по сопровождению программного продукта ориентироваться во внесенных в конфигурацию изменениях и оперативно ответить на поступивший вопрос пользователя [1].

Таким образом, возникает потребность в использовании системы структурированного хранения разработанных объектов конфигурации. Система должна выполнять следующие функции:

- отображать наименование решения, для которого выполнена доработка; вид объекта конфигурации; тип организации, для которой велась разработка; имя разработчика; дату завершения работы над объектом; комментарии; версию;
- структурировать объекты по типовым решениям, по сферам деятельности, по типу операции, которую разработка автоматизирует (например, учет больничных, расчет отпускных и т.д.);
- предоставлять возможность быстрого поиска по ключевым словам;
- предоставлять доступ к разработанным файлам организации только ограниченному кругу сотрудников для предотвращения утечки разработок конкурентам.

Проектирование архитектуры программной системы

Проектирование архитектуры программных систем начинается с анализа предметной области и выделения ролей [2]. Основные роли и их взаимодействие представлены на диаграммах вариантов использования (рис. 1, 2). Для администрирования программной системы отдельно выделена роль администратора (рис. 3).

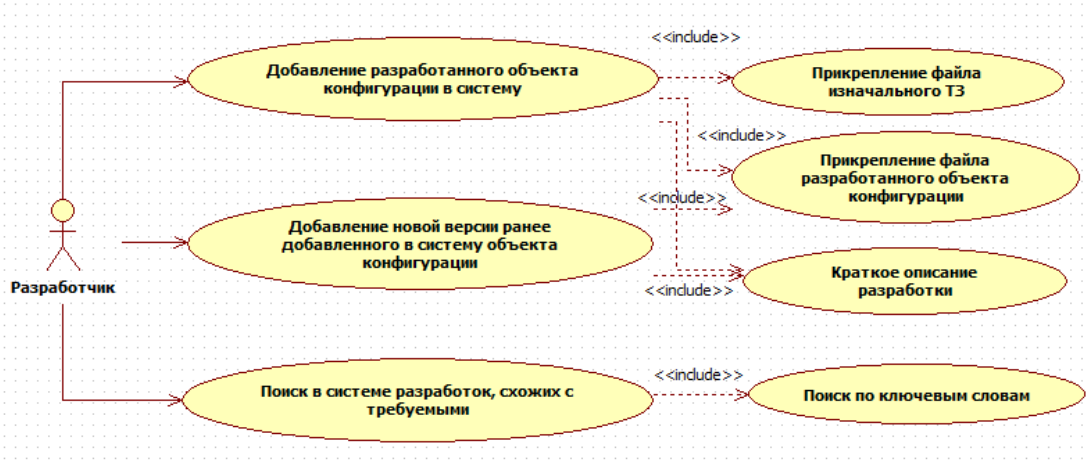


Рис. 1. Диаграмма вариантов использования программной системы администратором

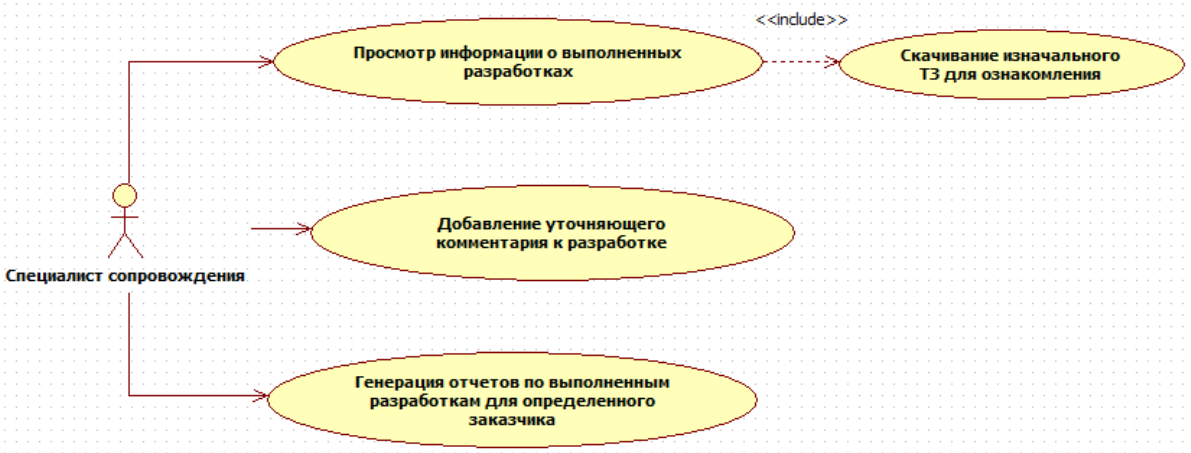


Рис. 2. Диаграмма вариантов использования программной системы специалистом информационно-технологического сопровождения

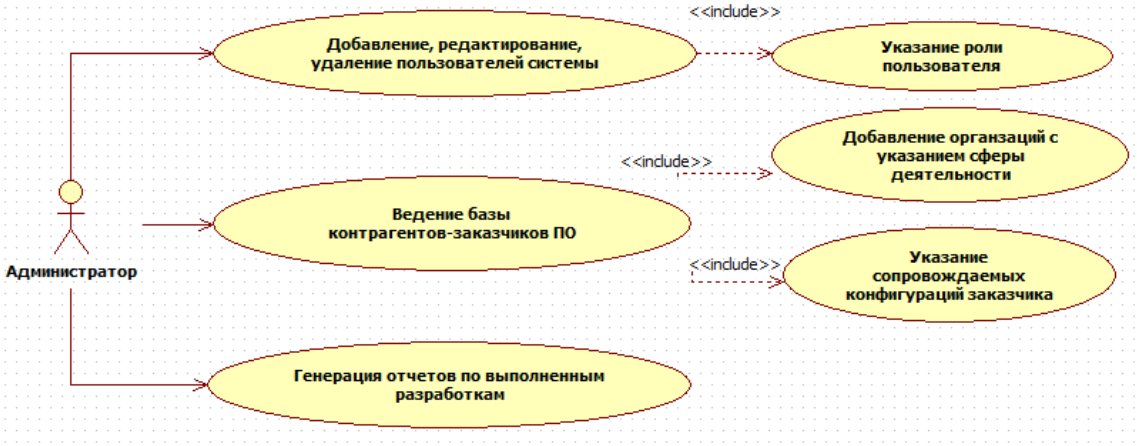


Рис. 3. Диаграмма вариантов использования программной системы администратором

Средства разработки

Для реализации программного обеспечения была выбрана платформа 1С:Предприятие 8.3 – программный продукт компании 1С, предназначенный для автоматизации деятельности на предприятии. Технологическая платформа «1С:Предприятие» представляет собой программную оболочку над базой данных. Имеет свой внутренний язык программирования, обеспечивающий, помимо доступа к данным, возможность взаимодействия с другими программами посредством OLE и DDE [3].

Платформа «1С:Предприятие» содержит такие инструменты для выполнения поставленных задач, как визуальное описание структур данных, написание программного кода, визуальное описание запросов, визуальное описание интерфейса, описание отчетов, отладка программного кода, профилирование. В ее составе: развитая справочная система, механизм ролевой настройки прав, инструменты создания дистрибутивов, удаленного обновления приложений, сравнения и объединения приложений, ведения журналов и диагностики работы приложения, создания Web-приложений и мобильных приложений, а также поддержка коллективной разработки, версионирования и пр. [4].

Разработка программного обеспечения

В разрабатываемой конфигурации были созданы такие объекты метаданных, как справочники, документы, перечисления, отчеты, регистры сведений.

Основные разработанные справочники:

-справочник «Файлы» - хранит наименования и файлы объектов конфигурации и файлы документации;

-справочник «Пользователи» - хранит информацию о ФИО пользователя, его должности и роли в системе, а также логин и пароль для авторизации;

-справочник «Организации» - предназначен для хранения информации о организациях, с которыми сотрудничает фирма;

- справочник «Сферы деятельности» - хранит сферы деятельности организаций-заказчиков ПО;

- справочник «Типовые конфигурации» - предназначен для хранения наименований типовых конфигураций, которые сопровождает фирма.

Документ «Карточка объекта конфигурации» предназначен для внесения и редактирования информации о разработанном объекте метаданных. В табличной части хранятся версии доработки и комментарии к ним.

Отчет «Список доработок типовой конфигурации заказчика» выводит таблицу всех выполненных для организации-заказчика доработок с указанием программиста и краткого описания предназначения этих доработок.

Разработанное приложение систематизирует и организует хранение ранее выполненных доработок.

Благодаря широким возможностям конфигурируемости, которые имеет разработанный программный продукт, конфигурация может быть легко доработана, если появятся новые потребности в систематизации хранения разработанных организацией объектов метаданных.

Библиографический список

1. Белов В.В., Крылова О.В. Обзор существующих средств структурированного хранения объектов конфигурации // Современные технологии в науке и образовании — СТНО-2018: сб. тр. международной науч.-техн. и науч.-метод. конф.: в 10 т. — Т. 4. — Рязань: Рязан. гос. радиотехн. ун-т, 2018. — С. 43–46

2. Белов В.В. Проектирование информационных систем – Москва: Издательство «Академия», 2013. – С.19–23.

3. Гладкий А.А., 1С: Управление торговлей 8.2. Настройка, конфигурирование, администрирование. – Москва: Мультимедийное издательство Стрельбицкого, 2018.

4. Кашаев С.М. Программирование в 1С:Предприятие 8.3. – Санкт-Петербург: Издательский дом «Питер», 2014.

УДК 004.043; ГРНТИ 20.53.19

СРЕДСТВА РАЗРАБОТКИ ОНТОЛОГИЙ. МЕТОДОЛОГИЯ IDEF5

Н.В. Куликов

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, ht_holod@mail.ru

Аннотация. В работе производится обзор существующих средств создания онтологий, их особенностей, достоинств и недостатков, производится их анализ и на его основе определяются требования для разрабатываемого средства создания онтологий в соответствии с методологией IDEF5.

Ключевые слова: IDEF5, OWL онтология, Protégé, Ontorion, семантическая сеть, методология, таксономия

ONTOLOGY DEVELOPMENT TOOLS. THE METHODOLOGY IDEF5

N.V. Kulikov

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, ht_holod@mail.ru

The summary. The paper reviews the existing tools for creating ontologies, their features, advantages and disadvantages, analyzes them and determines the requirements for the developed tool for creating ontologies in accordance with the IDEF5 methodology.

Keywords: IDEF5, OWL ontology, Protégé, Ontorion, semantic network, methodology, taxonomy.

Разработка систем обычно начинается с моделирования разрабатываемой системы. Модели можно представить в виде диаграмм. Существует стандарт IDEF, позволяющий изображать различные структуры, процессы, преобразования, классификации в виде диаграмм. IDEF (Integration Definition Metodology - Объединение Методологических Понятий) - семейство совместно используемых методов для процесса моделирования [1].

Стандарт IDEF содержит различные диаграммы для изображения процессов и структур. Среди них можно выделить стандарт IDEF5, который позволяет разрабатывать, изучать и поддерживать онтологию моделируемой системы.

На данный момент не существует средств для разработки диаграмм стандарта IDEF5. Однако онтологический подход все в большей степени внедряется в задачу разработки различных систем, семантические сети [2], используется в задачах искусственного интеллекта [3]. Цель данной работы – обзор средств для моделирования онтологий. Данный анализ будет использован для определения требований к разрабатываемой системе создания IDEF5 диаграмм.

Существует множество методов и средств для создания онтологий. Многие из них рассмотрены в [4]. В данной статье остановимся на некоторых.

Fluent Editor

Одним из средств разработки онтологий является программа Fluent Editor. Главное достоинство этого инструмента состоит в том, что он позволяет создавать онтологии на языке разработки онтологий OWL людям, не имеющим представления о синтаксисе OWL (хотя иметь представление о его концепциях, и о моделировании информационных структур все равно нужно). Онтологии создаются на естественном языке, Controlled English — то есть на обычном английском, к которому применены определенные правила и ограничения [5].

Для создания онтологии необходимо писать предложения на английском (немного формализованном). В качестве примера приведен скрипт для создания простой структуры предприятия, занимающегося модернизацией промышленного оборудования:

Every supplier **is** organization.
Every customer **is** organization.
Every technical-task **is a** document.
Every manual **is a** document.
Every customer create technical-task. **Every** supplier work-with technical-task.
Every customer have manual. **Every** supplier create manual.
Every customer have machine. **Every** cnc **is a** machine. **Every** dev-line **is** machine.

Каждое предложение определяет структуру разрабатываемой онтологии. Например, в первом предложении определяется класс «Organization» (организация) и его подкласс «Supplier» (поставщик). На русский данное предложение можно перевести следующим образом: «каждый поставщик является организацией». Ключевые слова в тексте онтологии в данной работе выделены жирным.

Помимо классов и подклассов, в структуре онтологии имеются отношения. Отношения задаются между классами с помощью глаголов. Если средство не может распознать глагол как ключевое слово или фразу, глагол автоматически становится отношением. Еще одна важная особенность касаемо глаголов это то, что система определяет формы глагола. То есть, если в предложении пишется не инфинитив (например, had или has), то в отношении все равно будет задан как инфинитив (have).

Ключевое слово «every» обозначает, что указываемый после него объект является подклассом класса, указанного после ключевого слова «is». Если «every» не использовать, то объект перед «is» будет объявлен как экземпляр класса. Например, если было бы необходимо объявить конкретную организацию-поставщика, можно было бы использовать следующее предложение:

Alphabet **is** supplier.

Таким образом, в структуру онтологии добавляется конкретная фирма.

Для задания атрибутов необходимо задать его хотя бы одному экземпляру. Например, у каждой организации в реквизитах указываются ИНН и ОГРН. Чтобы их задать организации поставщика, нужно записать следующий текст:

Alphabet inn **equal-to** 1234567890.
Alphabet ogrn **equal-to** 0987654321.

Имя атрибута задается перед ключевой фразой «equal-to», а после этой фразы указывается конкретное значение. Также может указываться конкретный тип атрибута. После ввода данных предложений будет создано два атрибута.

Также одним из важных правил синтаксиса языка является точка в конце предложения. Без нее система не распознает предложение и выдаст ошибку.

В конечном счете, классовая структура разработанной онтологии будет выглядеть так, как показано на рисунке 1.

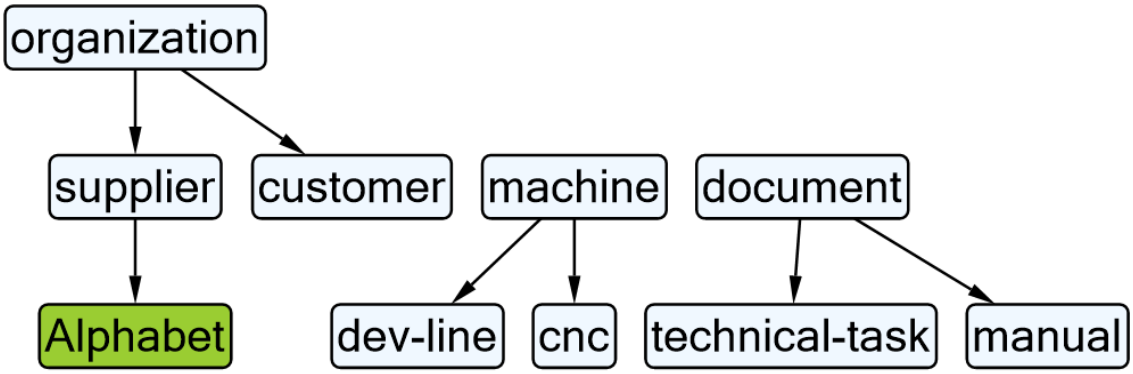


Рис. 1. Классификация в онтологии

Из достоинств средства можно выделить простоту для начинающего пользователя, не знакомого с языком OWL, малознакомому с онтологиями специалисту. Также система довольно наглядно представляет структуру, имеет средства импорта и экспорта в другие инструменты разработки онтологий.

Protégé

Другим средством для разработки онтологий является Protege. Это бесплатный редактор онтологий с открытым исходным кодом и система управления знаниями. Protege предоставляет графический пользовательский интерфейс для разработки онтологий. Помимо этого, средство включает в себя дедуктивные классификаторы для проверки согласованности моделей и вывода новой информации на основе анализа онтологии [6]. В отличии от предыдущего инструмента, Protégé имеет более широкий функционал.

Таксономия разделена по типу объектов: отдельно таксономия классов, отдельно для отношений, отдельно для свойств и отдельно для экземпляров.

Так же, как и в Fluent Editor, базовым классом является класс «thing». Структура таксономии совпадает с предыдущим примером.

Структуру онтологии можно вывести с помощью инструмента OntoGraph (рис. 2).

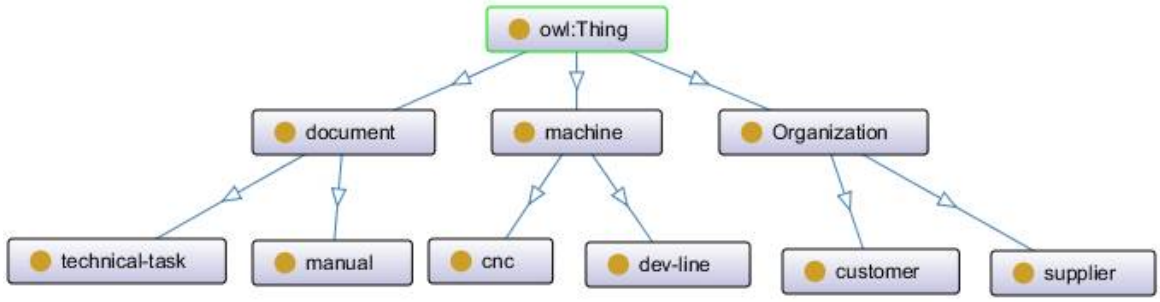


Рис. 2. Структура онтологии

Из достоинств средства можно выделить огромный спектр возможностей по созданию и редактированию онтологий. У данного средства существует версия, которую можно использовать онлайн прямо из браузера. Версия для устройств портативная и не нуждается в установке на компьютер.

Недостатки: сложность и запутанность, особенно для использования лицам, плохо знакомым с онтологиями.

Вывод

Рассмотренные средства разработки онтологий обладают рядом недостатков:

- сложность для понимания обычных пользователей. Рассмотренными решениями могут пользоваться лишь квалифицированные специалисты;
- отсутствие поддержки методологии IDEF5;
- отсутствие средств графического построения моделей. Рассмотренным решениям не хватает наглядности и легкости в изменении.

С учетом данных недостатков, видна необходимость средства, которое бы решало данные вопросы. Необходимо начать разработку такого средства, которое отвечало бы следующим требованиям:

- средство должно поддерживать методологию IDEF5;
- оно должно быть легким для использования непрофессионалами;
- должно быть наглядным, представлять собой графический редактор;
- поддерживать импорт в другие средства разработки онтологий.

Таким образом, в данной работе были рассмотрены средства для разработки онтологий, рассмотрена методология IDEF5 и определены требования к планируемой системе разработки онтологий на основе данной методологии.

Библиографический список

1. URL: http://idef.ru/idef.php (дата обращения 05.01.2020).
2. URL: https://elibrary.ru/item.asp?id=32597744 (дата обращения 05.01.2020).
3. Лазуткина В.С., Климов А.А., Куприяновский В.П., Намиот Д.Е., Покусаев О.Н. Онтологии больших данных, машинного обучения и искусственного интеллекта на цифровой железной дороге // International Journal of Open Information Technologies. 2019. №5. URL: https://cyberleninka.ru/article/n/ontologii-bolshih-dannyh-mashinnogo-obucheniya-i-iskusstvennogo-intellekta-na-tsifrovoy-zheleznoy-doroge (дата обращения: 05.01.2020).
4. Watrobski J. An Attempt to Knowledge Conceptualization of Methods and Tools Supporting Ontology Evaluation Process // University of Szczecin, Mickiewicza 64, 71-101 Szczecin, Poland. 2018.
5. Редактор онтологий на естественном языке: электронный портал Habr, 2013. URL: https://habr.com/ru/post/173015/ (дата обращения: 06.12.2019).
6. Protégé (software): открытая энциклопедия Wikipedia, 2019. URL: https://en.wikipedia.org/wiki/Protégé_(software) (дата обращения: 06.12.2019).

УДК 004.42; ГРНТИ 50.51.17

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ МЕТОДОЛОГИИ КАНБАН

А.С. Танцев

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, tochta4@gmail.com

Аннотация. В данной работе рассматривается разработка информационной системы, ориентированной на методологию Канбан. Описывается архитектурная модель Канбан-системы и ее компоненты.

Ключевые слова: методология Канбан, Канбан-доска, клиент-серверная архитектура, трехзвенная архитектурная модель, клиентский уровень системы, сервер, база данных, шаблон проектирования, «Модель-Представление-Контроллер».

KANBAN METHODOLOGY SOFTWARE

A.S. Tantsev

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, tochta4@gmail.com

The summary. This paper discusses the development of an information system focused on the Kanban methodology. The architectural model of the Kanban system and its components are described.

Keywords: Kanban methodology, Kanban board, client-server architecture, three-tier architectural model, client system level, server, database, design pattern, Model-View-Controller (MVC).

В современном мире существует необходимость оптимизации организационной деятельности предприятий, при этом возможны ситуации, когда обычных to-do листов не хватает. Поэтому в таких случаях требуется система, отслеживающая и показывающая всевозможные задачи: планирующиеся, находящиеся на стадии выполнения в данный момент, а также задачи, с которыми сотрудники кампании уже полностью справились.

Решением поставленной проблемы является методология управления разработкой Канбан, которая позволяет равномерно распределить нагрузку между работниками и реализовать принцип «точно в срок». А инструментом, который реализует данный функционал, является Канбан-доска [1].

Канбан-доска визуализирует рабочие процессы, что позволяет грамотно их проанализировать и повысить эффективность производства. Помимо этого Канбан-система позволяет сосредоточиться на приоритетах, благодаря чему сотрудники организации знают, какие задачи актуальны на данный момент и требуют незамедлительного решения, а какие могут подождать. Результатом использования данной системы является повышение эффективности управления задачами и оптимизация скорости и качества работ.

Канбан-система – это инструмент для совместной работы над проектами, использующийся в различных сферах: разработка ПО, маркетинг, строительство, логистика. Канбан-доска применима в любых отраслях, где присутствует постоянный поток задач. При этом стоит отметить, что данная система может использоваться в больших компаниях, но также в стартапах и даже для индивидуальных целей. Такая масштабируемость Канбан-системы и большое количество предметных областей, где ее можно применить, возникли благодаря способности данной системы подстраиваться под конкретную сферу деятельности любого размера [2].

Рассмотрим представление создаваемой информационной системы и ее компоненты.

Разрабатывая Канбан-систему, рационально использовать клиент-серверную архитектуру. При этом чтобы улучшить масштабируемость системы, будет использована многоуровневая архитектурная модель, в которой количество звеньев будет равно трем. А трехзвенная архитектура должна состоять из следующих компонентов системы: клиентский уровень, который подключен к серверу, соединенному с базой данных [3]. Описанную трех-

уровневую архитектурную модель можно представить на UML-диаграмме размещения, как показано на рисунке 1.

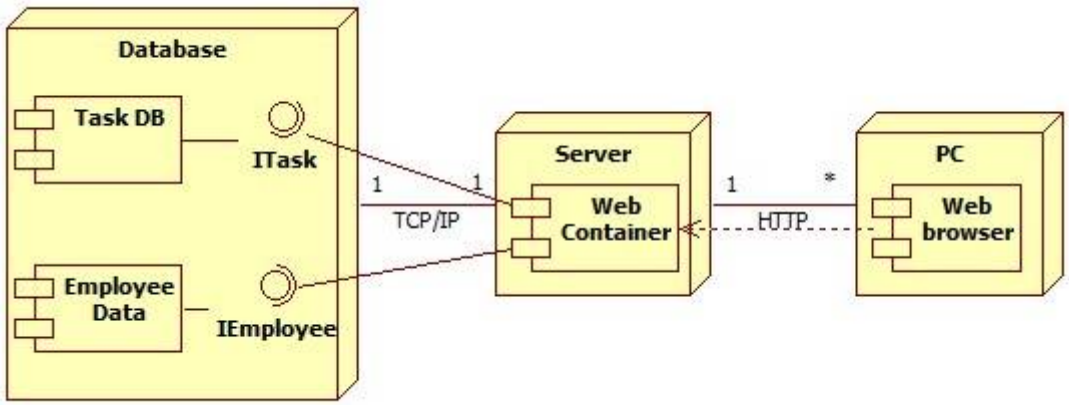


Рис. 1. UML-диаграмма размещения системы, реализующей методологию Канбан

Клиентская часть создаваемого приложения – это компонент, служащий для отображения данных Канбан-системы пользователю. При этом помимо самого интерфейса разрабатываемой системы в клиентской части будет находиться простейшая логика приложения: валидация данных, несложные операции с данными (сортировка, группировка). Следует отметить, что клиентский уровень не будет содержать основной бизнес-логики приложения, и он не будет иметь прямых связей с базой данных. Реализована же клиентская часть рассматриваемой архитектурной модели будет при помощи веб-клиента, так как его реализация едина и не зависит от используемого устройства. При этом Канбан-система будет состоять из четырех веб-страниц: домашняя страница, страница с выбранной доской, страница настройки выбранной доски и страница добавления задачи на выбранную доску. Реализация перечисленных выше веб-страниц будет осуществляться при помощи HTML, CSS и JS.

Серверная часть информационной системы – это компонент, который будет содержать в себе основную часть бизнес-логики. Уровень логики – это связующее звено между клиентским уровнем и уровнем данных, так как на сервере происходит обработка команд, поступающих из клиентской части системы и передача информации в базу данных. Серверная часть Канбан-системы будет реализована на программной платформе Java EE с применением сервлетов.

База данных системы, ориентированной на методологию Канбан – это компонент, который обеспечивает хранение данных. В качестве реализации уровня данных будет использоваться реляционная база данных, потому что она является самой распространенной и надежной. Данные в такой базе будут храниться в формате таблиц. А основными сущностями, которые будет хранить база данных, являются информация о сотрудниках компании и сведения о задачах, решаемые ими.

В качестве основы проектирования Канбан-системы будет применяться шаблон проектирования «Модель-Представление-Контроллер» (Model-View-Controller, MVC). Этот шаблон предполагает деление разрабатываемого приложения на три обособленные части:

• Модель хранит данные и выполняет команды контроллера.

• Представление определяет внешний вид информационной системы.

• Контроллер обрабатывает события, инициируемые пользователем, и уведомляет модель о необходимости изменений.

Важным достоинством шаблона «Модель-Представление-Контроллер» является то, что каждый из рассмотренных компонентов может быть модифицирован независимо от двух других, благодаря чему существенно облегчается расширение функциональности приложения.

Для реализации программного продукта предполагается применить следующие средства разработки: интегрированная среда разработки программного обеспечения IntelliJ IDEA, контейнер сервлетов Apache Tomcat, реляционная система управления базами данных (СУБД) Oracle.

Выбор интегрированной среды разработки обусловлен наличием бесплатной студенческой лицензии. Также факторами выбора послужило наличие таких функций как автоматическое форматирование кода, интеграция с различными инструментами сборки проекта (Ant, Maven, Gradle).

В качестве контейнера сервлетов был выбран Apache Tomcat, потому что он является бесплатным инструментом с открытым исходным кодом.

Из существующих реляционных СУБД была выбрана Oracle, благодаря ее высокой производительности и простоте в использовании. К тому же Oracle – это самая популярная система управления базами данных на текущий момент.

В итоге в результате разработки информационной системы, реализующей методологию Канбан, должны быть реализованы следующие основные элементы и функции [4]:

• Домашняя страница.
  – Список созданных досок.
  – Создание задачи на доску.
  – Создание новой доски.
• Страница выбранной доски.
  – Список фильтров.
  – Колонки, с помощью которых определяется статус задач на данный момент.
  – Приоритеты задач: высокий, средний, низкий.
  – Иконки сотрудников около решаемых ими задач.
• Страница настройки выбранной доски.
  – Список колонок на доске.
  – Добавление и удаление колонки.
  – Список текущих фильтров.
  – Добавление и удаление фильтра.
• Страница добавления задачи на выбранную доску.
  – Название, описание и приоритет задачи.
  – Данные о сотруднике, который будет работать над создаваемой задачей.
  – Список досок, определяющий, для какой доски создается задача.
  – Создание задачи.

Текущими результатами работы являются:

• Исследование предметной области создаваемой информационной системы.
• Рассмотрение архитектуры системы и ее компонентов.
• Перечисление элементов и функций, которые должны быть реализованы при разработке Канбан-системы.
• Описание разработки программного обеспечения информационной системы, реализующей методологию Канбан.

Библиографический список

1. Андерсон Д. Канбан. Альтернативный путь в Agile; пер. с англ. А. Коробейникова. – М.: Манн, Иванов и Фербер, 2017. – 335 с.
2. Книберг Х., Скариан М. Scrum и Kanban: выжимаем максимум. – Киев: InfoQ, 2010. – 78 с.

3. Фаулер М. Архитектура корпоративных программных приложений. – М.: Издательский дом "Вильямс", 2006. – 544 с.

4. Танцев А. С. Разработка информационной системы, реализующей методологию канбан // Новые информационные технологии в научных исследованиях: материалы XXIV Всероссийской научной-технической конференции студентов, молодых ученых и специалистов. Рязань: ИП Коняхин А.В. (Book Jet), 2019. Т. 2. С. 21-22.

УДК 004.42; ГРНТИ 50.41.25

РАЗРАБОТКА КРОССПЛАТФОРМЕННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ВЕДЕНИЯ РАБОЧЕГО ГРАФИКА СОТРУДНИКОВ

Е.Е. Перевезенцев

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, gilza111@mail.ru

Аннотация. В работе рассматриваются аспекты программной реализации информационной системы для учета и анализа рабочего графика предприятий. Конкретизируется формат разработки, обосновывается выбор инструментов, которые будут использоваться при разработке и описываются принципы программирования, которых планируется придерживаться при реализации информационной системы.

Ключевые слова: Android, Java, ООП, SOLID.

DEVELOPMENT OF CROSS-PLATFORM SOFTWARE FOR MANAGING EMPLOYEES' WORK SCHEDULES

E.E.Perevezentsev

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, gilza111@mail.ru

The summary. The paper considers aspects of software implementation of the information system for accounting and analysis of the working schedule of enterprises. The development format is specified, the choice of tools that will be used in the development is justified, and the programming principles that are planned to be followed when implementing the information system are described.

Keywords: Android, Java, OOP, SOLID.

При реализации бизнес-проекта, после того как выявлены экономические факторы, влияющие на разработку и выделена целевая аудитория, необходимо четко определить программные аспекты будущего приложения.

Прежде всего, следует конкретизировать формат будущей разработки, выбрать системы, для которых это будет реализовано и инструменты, которые помогут создать конечный продукт.

Для удобного взаимодействия сотрудников с рабочим графиком следует реализовать систему в формате мобильного приложения. В начале 2020 года доля смартфонов на базе Android превысила 80%. Выбрав Android как систему, для которой будет написано приложение, удастся охватить бо̀льшую часть аудитории.

Так как выбор системы пал на Android необходимо писать программу на языке программирования Java или Kotlin. На это есть веская причина.

Android - это Unix подобная операционная система, а если быть точным, то это Linux, но переделанный как, например, Ubuntu или Debian [1]. При этом производители смартфонов, используя AOSP - проект Android с открытым исходным кодом, делают свои надстройки над Android. Примеры таких надстроек: One UI (Samsung), EMUI (Huawei, Honor), ColorOS (Oppo, Realme), MIUI (Xiaomi), FuntuchOs (Vivo), pixel experience (Google), OxygenOS (OnePlus) [2]. Одни системы близки к AOSP, другие сильно от него отличаются (например, EMUI от Huawei) . Самый важный вывод из этого в том, что в AOSP входит ядро Linux и на-

тивный язык Java Machine, следовательно, кроссплатформенность без проблем будет обеспечена за счет Java Machine. Новый Kotlin работает на том же движке [3].

Решающую роль в выборе между Kotlin и Java играет вопрос удобства и количества необходимых библиотек. Принято решение использовать языка Java, так как он гораздо старше и стабильнее Kotlin.

Исходя из личных предпочтений, в качестве среды разработки была выбрана Intellij IDEA. Выбор был сделан по причине ряда преимуществ и удобств Intellij IDEA, а именно:

- Поддержка нескольких систем сборки;
- Пользовательский интерфейс тестового запуска приложений;
- Поддержка Google App Engine, Grails, GWT, Hibernate, Java EE, OSGi, Play, Spring, Struts и других;
- Встроенные средства развертывания и отладки для большинства серверов приложений;
- Интеллектуальные текстовые редакторы для HTML, CSS и Java;
- Интегрированный контроль версий;
- AIR Mobile с поддержкой Android и iOS [4].

Также в Intellij IDEA есть возможности добавлять язык Kotlin в код Java в виде отдельных классов, среда допускает это.

Вышеперечисленные рассуждения привели к необходимости использовать ООП – объектно-ориентированное программирование. На мой взгляд, суть всего ООП – использование шаблонов. Шаблоны - это понятие, которое подразумевает использование определенного подхода или алгоритма, который уже существует для решения проблемы в той или иной ситуации. Шаблоны удобны для решения типичных проблем и написания чистого кода. В моем понимании, чистый код - это максимально простой код без повторений. Также паттерны позволяют написать масштабируемую программу.

Принципы чистого кода - это принципы объектно-ориентированного программирования SOLID. Паттерны в совокупности с SOLID позволяют написать максимально комфортный и чистый код, что позволит расширять и изменять приложение.

Один из аспектов реализации информационной системы – это архитектура системы. Была выбрана модульная архитектура с целью комфортной замены модулей при адаптации системы под разных заказчиков с минимальными затратами ресурсов.

Программная составляющая будущего приложения будет содержать следующие части:

- серверная часть, где будут храниться и фиксироваться данные сотрудников, время работы и тд;
- клиентская часть, где необходимо реализовать набор функций, предоставляемый системой и обеспечить удобство для пользователя, которое заключается в максимально быстром освоении интерфейса и возможности реализовать функцию за минимальное число кликов и свайпов;
- сетевой код - это связь клиентской и серверной части и отправка пакетов на сервер и обратно. Серверная часть будет представлена наиболее простой базой данных.

Серверная часть будет состоять из SQL сервера и надстроек над ним, которые будут представлять основные функции системы. Реализовать надстройки планируется с использованием шаблона «Адаптер».

Клиентская часть будет состоять из кнопок и таблицы с учетом удобства их просмотра и кликов. Планируется, что каждая кнопка и каждый режим работы будут реализованы отдельным классом с применением паттернов и принципов SOLID. За основу будет взят архитектурный шаблон MVC (Модель-Вид-Контроллер) [6].

Планируется создать два интерфейса: один для веб браузеров, другой для смартфонов.

Интерфейс для веб браузеров слева будет предоставлять доступ к функциональности системы и к переходам между страницами. Сверху экрана планируется разместить элементы с доступом к основным функциям каждого пункта.

В интерфейсе для смартфонов функции каждого пункта будут располагаться снизу.

Различия в расположении элементов в интерфейсах для веб браузеров и смартфонов обусловлено тем, что при использовании смартфона управление происходит большим пальцем. При расположении функций внизу экрана пользователю не придется тянуться наверх, что довольно сложно с размера современных смартфонов.

Поведенческие и структурные шаблоны станут вспомогательными при разработке [7]. Я отказываюсь от использования порождающих шаблонов, так как их удобство в настоящее время под вопросом.

В сетевом коде есть несколько проблем:

1. Ресурсы. Эта проблема связана с тем, какие данные передавать пользователю при условии, что они у него уже есть, и как это сделать максимально эффективно;
2. Ссылки на объекты. Эта проблема завязана на том, как заставить пользователей обрабатывать определенные объекты;
3. Дельта-компрессии. Эта проблема связана с аппаратной частью, а именно с пропускной способностью сервера. Если мне нужно передавать информацию 100 раз за 1 секунду, то это, допустим, 22 мб/с в зависимости от того что передается. Дельта-компрессия решает эту проблему лишь отчасти. Она предает только изменения, что сокращает траты на пропускную способность.

В итоге, при решении этих проблем, ища компромиссы и глядя на опыт прошлых лет, можно реализовать задуманное приложение максимально эффективно по затратам ресурсов, так как большая часть уже придумана, необходимо лишь воспользоваться этим, а библиотеки Java и SQL дают такую возможность.

Таким образом, после написания статьи были получены следующие итоги:

1. Выбрана система, для которой будет разработана система;
2. Выбран язык программирования для написания системы;
3. Выявлены преимущества среды разработки Intellij IDEA;
4. Аргументирована необходимость использования объектно-ориентированное программирование, паттерны и принципы SOLID;
5. Определены части, из которых будет состоять информационная система и уточнены детали реализации каждой из частей;
6. Описаны проблемы, которые могут помешать качественной разработке информационной системы.

Библиографический список

1. Негус К., Каэн Ф. Ubuntu и Debian Linux для продвинутых: [более 1000 незаменимых команд : пер. с англ.] – Питер, 2010. – 347 с.
2. https://lifehacker.ru/operacionnye-sistemy-android/
3. Скин Джош, Гринхол Дэвид, Kotlin. Программирование для профессионалов – Спб.: Питер, 2020. – 464 с.: ил. – (Серия «Для профессионалов»).
4. www.internet-technologies.ru%2Farticles%2F10-luchshih-ide.html#header-10606-5
5. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж., Приемы объектно-ориентированного программирования. Паттерны проектирования – Спб.: Питер, 2015. – 368 с.: ил. – (Серия «Библиотека программиста»).
6. https://refactoring.guru/ru/design-patterns/catalog

УДК 004.42; ГРНТИ 50.41.23

АВТОМАТИЗАЦИЯ УЧЕТА ПРОГРАММНОГО И АППАРАТНОГО ОБЕСПЕЧЕНИЯ КОМПЬЮТЕРОВ ПОДРАЗДЕЛЕНИЯ УЧРЕЖДЕНИЯ

А.Ю. Голованов

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, esbodev@gmail.com

Аннотация. В работе проектируется программное обеспечение информационной системы для автоматизации учета наличия и работоспособности программного и аппаратного обеспечения компьютеров. Обосновывается тип архитектуры системы. Проектируются все составляющие архитектуры.

Ключевые слова: автоматизация учета, учет компьютеров, трехуровневая архитектура, сервлеты, Java EE, Front controller, Oracle Database.

AUTOMATION OF SOFTWARE AND HARDWARE ACCOUNTING IN INSTITUTION DIVISION

A.Y. Golovanov

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, esbodev@gmail.com

The summary. The paper discusses designing of an information system software designed for automation of software and hardware presence and operability accounting. Architecture is being justified. All architecture components are being designed.

Keywords: accounting automation, computer accounting, three-tier architecture, servlets, Java EE, Front controller, Oracle Database.

Проектируемая информационная система (ИС) разрабатывается для кафедры университета. На рынке не имеются бесплатные альтернативы, удовлетворяющие необходимым требованиям. Следовательно, актуальность работы оправдана. ИС должна выполнять следующие задачи:

1) предоставление инструментов для формирования и обработки запросов на исправление проблемы с компьютером;

2) хранение информации о пользователях системы;

3) предоставление инструментов для администрирования пользователей.

Пользователи системы делятся на следующие группы:

1) администраторы;

2) создатели запросов (студенты и преподаватели);

3) инженеры;

4) проверяющие.

Администраторы занимаются добавлением, удалением пользователей, а также обновлением информации о них в системе. Создатели запросов, а именно студенты и преподаватели, имеют возможность формирования запросов на решение проблем с компьютером. Инженеры имеют возможность изменения состояния запроса на «Проблема подтверждена», «Проблема отсутствует» и «Проблема предварительно решена». Проверяющие имеют возможность изменять состояние запроса на «Подтверждено решение проблемы» и «Проблема решена неверно».

В качестве архитектуры ИС рационально выбрать трехуровневую архитектуру [1]. При использовании такой архитектуры система будет разделена на 3 части:

1) уровень данных;

2) уровень логики;

3) уровень клиента.

Рассмотрим каждую из трех частей.

Уровень данных

Уровень данных представляет собой реляционную базу данных. Реализован данный уровень с использованием СУБД Oracle Database.

В базе данных хранятся следующие таблицы.

Таблица пользователей. В данной таблице должна находиться основная информация о человеке, которому принадлежит учетная запись. К такой информации относится фамилия, имя, отчество пользователя. Также каждый пользователь имеет уникальный идентификатор. Так как в ИС предусмотрена система авторизации, здесь также должна храниться пара логин-пароль. В качестве логина будет использоваться электронная почта пользователя. Пароль в открытом виде храниться не будет, т.к. иначе, в случае несанкционированного доступа к БД, злоумышленник сможет получить пароль пользователя, который может совпадать с паролем на другом ресурсе. Поэтому, для защиты данных пользователя, пароль будет храниться в виде хеша. В качестве алгоритма хеширования был выбран алгоритм BCrypt [3], широко использующийся для хеширования паролей. Пользователи системы разделены на группы, следовательно, в таблице пользователей разумно также хранить идентификатор группы пользователя.

Таблица групп пользователей. В этой таблице содержится список групп пользователей с их идентификаторами и названиями.

Таблица прав групп пользователей. В данной таблице содержится список прав групп пользователей без привязки к конкретным группам. Атрибуты таблицы: идентификатор права, название права.

Таблица прав конкретных групп пользователей. Эта таблица обеспечивает связь многие ко многим между таблицей групп пользователей и таблицей прав групп пользователей. Атрибуты таблицы: идентификатор группы, идентификатор права группы.

Таблица заявок на исправление проблем. Данная таблица содержит список заявок на исправление проблемы с компьютером. Каждая заявка имеет уникальный идентификатор, что должно быть отражено в таблице. Заявку отправляет пользователь системы, следовательно, также необходимо хранить идентификатор отправителя. У заявки будет присутствовать краткое и подробное описание проблемы. Заявки будут категоризированы по типу проблемы и приоритету. По мере решения проблемы, заявки меняют свое состояние, которое также будет отражено в таблице.

Таблица категорий заявок. В данной таблице хранится список категорий заявок на исправление проблемы с компьютером. Атрибуты: идентификатор заявки, название заявки.

Таблица состояний заявок. Здесь хранится список состояний заявок на исправление проблемы с компьютером. Атрибуты таблицы: идентификатор состояния, название состояния.

Таблица приоритетов заявок. В этой таблице содержится список приоритетов заявок на исправление проблемы с компьютером. Атрибуты таблицы: идентификатор приоритета, имя приоритета, важность приоритета. Важность приоритета является целочисленным числом. Чем больше это число, тем более важным считается приоритет.

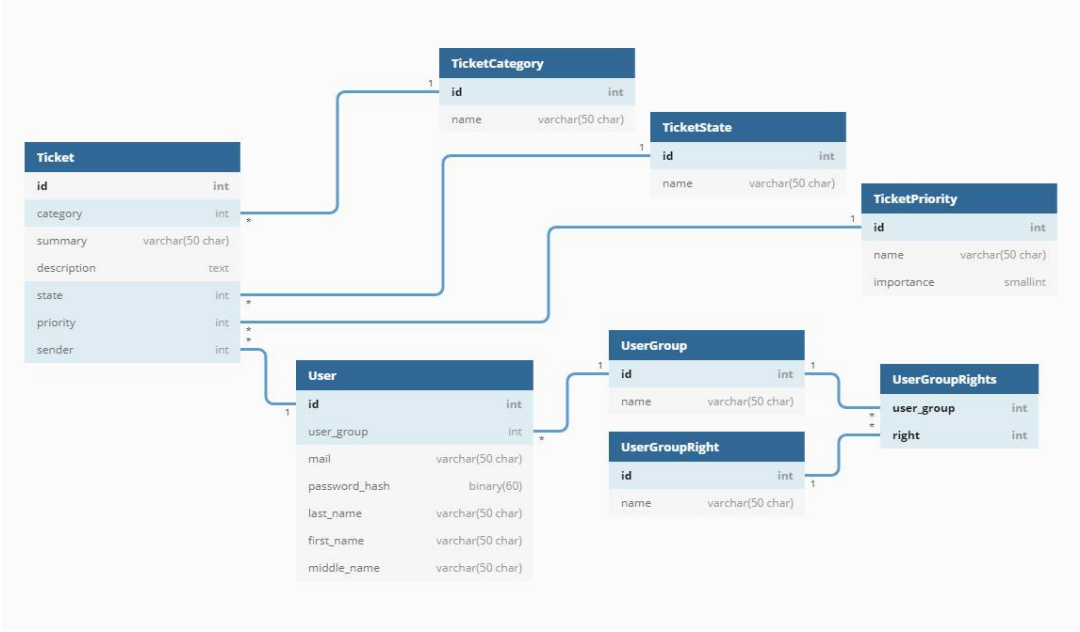


Рис. 1. Диаграмма таблиц БД

Уровень логики

На уровне логики используется HTTP сервер и контейнер сервлетов Apache Tomcat. В контейнере сервлетов будет содержаться один сервлет, который будет реализован в соответствии с шаблоном проектирования Front controller [2]. При обращении клиента к серверу, контейнер сервлетов, в соответствии со свойствами http-запроса, создаст объекты запроса и ответа, которые передаст цепочке фильтров. В случае, если запрос будет корректным, и, соответственно, не отфильтруется, управление передастся сервлету. Сервлет, в свою очередь, в зависимости от указанного в параметрах запроса действия, делегирует обработку запроса соответствующему методу команды, отвечающей за указанное действие.

Связь между уровнем логики и уровнем данных обеспечивается реализацией шаблона DAO [2]. Данный шаблон позволяет использовать унифицированный интерфейс для доступа к различным базам данных, что, в свою очередь, позволит в будущем мигрировать с одной базы данных на другую, не переписывая при этом большое количество кода.

Представление ИС в данной архитектуре представлено на уровне логики в виде jsp-страниц. Ниже представлен список страниц с их описанием.

Страница авторизации. Изначальная страница приложения. Любой неавторизованный пользователь системы переадресуется сюда. На этой странице содержатся поля ввода логина и пароля, а также кнопка входа. После успешной авторизации пользователь переадресуется на страницу с меню доступных действий.

Меню доступных действий. Здесь отображаются все действия, доступные пользователю. При выборе действия осуществляется переход на соответствующую страницу.

Страница добавления запроса на решение проблемы с компьютером. Данная страница доступна создателям запросов. На странице находятся кнопка добавления запроса и следующие поля ввода:

1) категория проблемы;
2) краткое описание проблемы;
3) подробное описание проблемы.

Страница просмотра запросов на решение проблемы с компьютером. Доступна создателям запросов, инженерам и проверяющим. У инженеров и проверяющих также присутствует возможность изменения состояния запроса.

Страница редактирования списка пользователей. Доступна администраторам. Содержит список пользователей и инструменты для их добавления, удаления и изменении информации о пользователях.

Уровень клиента

Клиентский уровень ИС будет спроектирован как «тонкий» веб-клиент. Под веб-клиентом подразумевается, что в качестве клиента будет использоваться веб-браузер, передающий информацию от пользователя к серверу в виде get или post запроса и отображающий ответ сервера в виде веб-страницы в формате html или xhtml. «Тонкий» означает, что клиент будет получать готовый html с сервера, не проводя над ним никаких преобразований.

Заключение

Были сформулированы задачи ИС, выделены группы пользователей системы. Была обоснована архитектура ИС и предварительно спроектирована каждая из ее частей. В ходе дальнейших исследованиях будет спроектирована реализация функциональности системы.

Библиографический список

1. Молчанов А.Ю. Системное программное обеспечение. – Санкт-Петербург: Издательский дом «Питер», 2010.

2. Блинов И.Н., Романчик В.С. Java. Методы программирования. – Минск: Издательство «Четыре четверти», 2013.

3. Джульен В. Безопасный DevOps. Эффективная эксплуатация систем. – Санкт-Петербург: Издательский дом «Питер», 2019.

УДК 004.42; ГРНТИ 20.53

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТА ЦИТИРОВАНИЙ ПУБЛИКАЦИЙ СОТРУДНИКОВ ОБРАЗОВАТЕЛЬНЫХ И НАУЧНЫХ УЧРЕЖДЕНИЙ

В.В. Ромашкова

Рязанский государственный радиотехнический университет имени В.Ф. Уткина,
Российская Федерация, Рязань, romashkova720@gmail.com

Аннотация. В статье обоснована необходимость реализации системы учета наукометрических показателей учебных и научных учреждений в виде веб приложения, приведены достоинства веб приложения, описана его архитектура. Рассказано о программном обеспечении, которое необходимо для реализации системы.

Ключевые слова: веб приложение, веб сервер, сервлет, Java, JDBC.

DEVELOPMENT OF AN INFORMATION SYSTEM FOR ACCOUNTING CITATIONS OF PUBLICATIONS OF EMPLOYEES OF EDUCATIONAL AND SCIENTIFIC INSTITUTIONS

V.V. Romashkova

Ryazan State Radio Engineering University named after V.F. Utkin,
Russia, Ryazan, romashkova720@gmail.com

The summary. The article substantiates the need to implement a system of accounting for scientometric indicators of educational and scientific institutions in the form of a web application, shows the advantages of a web application, and describes its architecture. It is described about the software that is necessary for the implementation of the system.

Keywords: web application, web server, servlet, Java, JDBC.

В работе [1] был сделан вывод, что производить учет вручную путем поиска профилей студентов и сотрудников учреждений в электронной научной библиотеке, просматривать и заносить интересующую информацию в базу данных занимает много времени и может приводить к ошибкам по объективным и субъективным причинам. Поэтому, следует автоматизировать работу, создав приложение, которое позволит уменьшить вероятность ошибки и сократить время на выполнение задач.

Реализовать систему планируется в виде веб-приложения. Веб-приложение — это клиент-серверное программное обеспечение, где в качестве клиента выступает браузер, который отображает пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него. [2]. Достоинствами веб-приложения являются [3]:

- Отсутствие необходимости устанавливать дополнительное программное обеспечение. Все что требуется для полноценной работы – это браузер, зачастую поставляемый вместе с операционной системой, и доступ в интернет;
- Отсутствие требований к ресурсам компьютера и аппаратной платформе;
- Поддержка старых версий программ и обратная совместимость. У веб-приложений существует лишь одна версия, в которой работают все пользователи, и в случае выхода новой версии все без исключения автоматически переходят на нее, порой даже не замечая этого;
- Веб-приложения позволяют пользователя быть мобильными, предоставляя возможность работать в сети с любого устройства, сохраняя результаты работы на сервере.

Из выше перечисленных достоинств можно сделать вывод, что веб-приложение отлично подходит для организации учета наукометрических показателей и комфортной работы пользователей с информацией.

Веб-приложение состоит из клиентской и серверной частей.

Клиентская часть включает в себя пользовательский интерфейс, через который пользователь взаимодействует с сервером, формируя, посылая запросы и получая ответы на них.

Серверная часть, в свою очередь, может быть представлена двумя составляющими: веб-сервером, который реализует бизнес-логику приложения, и системой управления базой данных (СУБД), которая реализует управление данными.

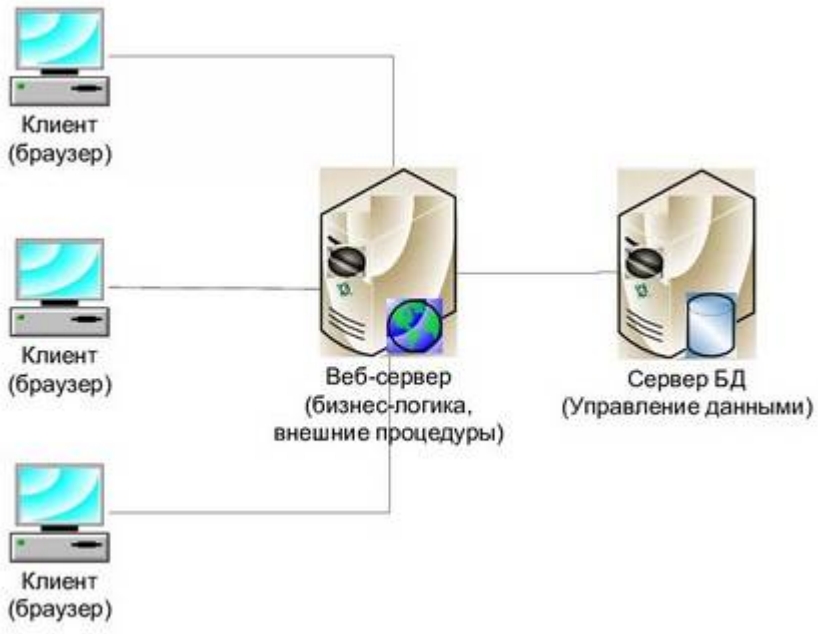


Рис. 1. Архитектура веб-приложений

Клиентская часть будет реализована на языке программирования Java. Для создания графического интерфейса планируется использовать такие инструменты как HTML и CSS.

Веб-сервер — это сервер, который принимает HTTP-запросы от клиентов и выдает им HTTP ответы. Планируется использовать контейнер сервлетов Apache Tomcat [4], также написанный на Java, так как он поддерживает технологию сервлетов.

Сервлет - центральная часть любых веб-приложений на Java. Сервлет - это интерфейс Java, реализация которого расширяет функциональные возможности сервера. Веб-сервер имеет специальный конфигурационный файл, который ему сообщает о том, какой сервлет надо выполнить в случае прихода определенного URL, который содержится в HTTP-запросе Именно с помощью сервлетов в Java строится клиент-серверная архитектура [5].

Сервером базы данных выступит Oracle SQL Developer.

Для реализации веб-приложения была выбрана Java, потому что ее неотъемлемая часть - технология сервлетов - имеет ряд значительных преимуществ [6]:

- Сервлет работает быстрее, чем CGI - стандарт интерфейса, используемого для связи внешней программы с веб-сервером, поскольку он не включает создание нового процесса для каждого нового полученного запроса;
- Сервлеты, написанные на Java, не зависят от платформы;
- Устраняет накладные расходы на создание нового процесса для каждого запроса, поскольку сервлет не запускается в отдельном процессе. Существует только один экземпляр, который обрабатывает все запросы одновременно. Это также экономит память и позволяет сервлету легко управлять состоянием клиента;
- Это серверный компонент, поэтому сервлет наследует безопасность, обеспечиваемую веб-сервером;
- Программный интерфейс приложения (API), разработанный для сервлета Java, может использовать широкий спектр других API-интерфейсов, созданных на Java, например, таких как JDBC (Java DataBase Connectivity), для доступа к базе данных.

Для организации взаимодействия между приложением и СУБД будет использоваться стандартный прикладной интерфейс Java – JDBC. Взаимодействие осуществляется с помощью драйверов JDBC, обеспечивающих реализацию общих интерфейсов для конкретных СУБД и конкретных протоколов [7].

В заключении, подведем итоги, изложенные в статье:

1. Выбран формат разработки информационной системы
2. Приведены достоинства реализации информационной системы в формате веб-приложения
3. Описана архитектура веб-приложения
4. Выбраны язык программирования, технологии разработки и программное обеспечение
5. Приведены достоинства реализации веб-приложения на Java с использованием сервлетов

Библиографический список

1. Ромашкова В. В., Учет наукометрических показателей учебных и научных учреждений // Новые информационные технологии в научных исследованиях: материалы XXIV Всероссийской научно-технической конференции студентов, молодых ученых и специалистов; Рязань: ИП Коняхин А.В. (Book Jet), 2019. – с.130-131

2. Иконников М. А., Карманов И. Н. Меры и требования к защищенным веб-приложениям // Интерэкспо ГЕО-Сибирь. XV Международный научный конгресс, 24–26 апреля2019 г., Новосибирск [Текст]: сборник материалов в 9 т. Т. 6: Магистерская научная сессия «Первые шаги в науке»; Новосибирск: СГУГиТ, 2019. – 261 с.

3. Вылегжанина А. О., Информационно-техническое и программное обеспечение управления проектом : учебное пособие / А. О. Вылегжанина. – М. – Берлин: Директ-Медиа, 2015. – 429 с.

4. http://tomcat.apache.org/
5. Машнин Т. С., Современные Java-технологии на практике. – Спб.: БХВ-Петербург, 2010. – 560 с.
6. http://espressocode.top/introduction-java-servlets/
7. Блинов И. Н., Романчик В. С. Java. Методы программирования: учебное-методическое пособие. – Минск: издательство «Четыре четверти», 2013. – 896 с.

УДК 621.396

РАЗРАБОТКА СЕНСОРНОЙ СИСТЕМЫ УПРАВЛЕНИЯ ИНФРАСТРУКТУРОЙ УСТАРЕВШИХ И ИНТЕЛЛЕКТУАЛЬНЫХ МУНИЦИПАЛЬНЫХ СИСТЕМ

Е.В. Чайко, А.Б. Абдуллаева, Е. Зайцев, А. Жунусов, Т. Абишева
Рижский технический университет институт промышленной электроники и электротехники
Латвия, Рига, Jelena.Caiko@rtu.lv

Аннотация. В данной работе представлена разработка общей парадигмы управления на основе беспроводных сенсорных сетей, которая учитывает внедрение сети LoRa WAN в Латвии. Эта работа представляет собой исследование по созданию интеллектуальных муниципальных систем в локальном облаке сервисов, использующих SOA и IoT. Брокер, который применяет сервисы оркестровки, находится на шлюзе, который обеспечивает функции преобразования адаптера и протоколов, и на инструменте для соединения аппаратных устройств. Мы внедрили сервис-брокер с поддержкой MQTT, который использует инструмент программирования потока данных Node-RED для соединения разнородных аппаратных устройств и узлов и API-интерфейсы для онлайн-сервисов. В данной работе представлен и обсужден пример использования сервис-брокера для управления муниципальными системами в городской среде.
Ключевые слова: промышленные информационные системы, техническое обслуживание, стандартизация, беспроводные датчики.

DEVELOPMENT OF SENSOR SYSTEM FOR LEGACY AND SMART MUNICIPAL SYSTEMS INFRASTRUCTURE CONTROL

J.V. Caiko, A.B. Abdullayeva, J. Zaicev, A. Zunusov, T. Abisheva
The Riga Technical University Institute of Industrial Electronics and Electrical Engineering
Riga, Latvia Jelena.Caiko@rtu.lv

The summary. This work presents the development of a common control paradigm based on wireless sensor networks, which takes into account implementation of LoRa WAN network in Latvia. This work presents a research on creation of smart municipal systems in a local cloud of services that apply SOA and IoT. A broker that applies orchestration services resides on a gateway, which provides adapter and protocols translation functions, and on a tool for wiring together hardware devices. We implemented a MQTT enabled service broker, which deploys data flow programming tool Node-RED for wiring together divergent hardware devices and nodes, and APIs for online services. A case study of the service broker implemented for control of municipal systems in urban environment is presented and discussed in this work.
Keywords: Industrial information systems, Maintenance, Standardization, Wireless sensors.

Управление различными сетями инфраструктуры беспроводными сенсорными сетями (БСС) в сочетании с Интернетом вещей (IoT) является новой тенденцией развития всего промышленного сообщества, и Латвия является одним из лидеров среди стран ЕС. Маломощные глобальные сети LPWAN или LoRaWAN сети внедрены в Нидерландах, Бельгии, Франции, Венгрии, Эстонии, США, Южной Корее и в других местах. Лидер на рынке телекоммуникаций и ИТ провайдер в Латвии Lattelecom работал вместе с Рижским техническим университетом для развертывания по всей стране сети LoRa WAN. Сеть была успешно внедрена в Риге, где полный охват будет развернут на первом этапе проекта с целью сделать столицу Латвии одним из самых «умных» городов в Европе. Более широкое внедрение будет

продолжено на основе интересов городских властей и предприятий с национальным охватом планируется завершить к концу 2018 года.

Источник питания для беспроводных сенсорных сетей, особенно для наружных применений, в диапазоне дальности действия источника питания может быть реализован с помощью интегрированного решения по источнику питания, которое может питать всю сенсорную сеть для увеличения надежного рабочего времени БСС.

Архитектурная эталонная модель IoT [1] определяется европейским проектом IoT-A, целью которого является решение всех связанных с IoT проблем. IoT-A предоставляет модели высокого уровня для определения эталонной архитектуры.

Проект Arrowhead определяет основу для создания распределенных промышленных систем совместными сетевыми встроенными устройствами [2], [3]. Эта структура предназначена для интеграции популярных протоколов уровня приложений с использованием уникальной сервис-ориентированной архитектуры (COA).

Различные исследователи используют традиционные подходы к распределенным системам, такие как общие методы передачи сообщений или обмена данными [13]. Модель макропрограммирования, которая способна контролировать сеть распределенных датчиков в целом, а не программировать отдельные элементы, предложена в [14]. Giang [15] и Blackstok и др. [16] предложили расширенную версию модели потока данных, чтобы выразить логику приложения в настройке Smart City, которая называется «Адаптивный распределенный поток данных», как более подходящая для крупномасштабного IoT.

Для практической реализации сервисов оркестровки мы рассмотрели несколько систем потоков данных, таких как Blockly [17], [18], клиентская библиотека JavaScript для создания визуального блочного языка программирования и редакторов. Это проект Google с открытым исходным кодом под лицензией Apache 2.0. Другая система - Domoticz [19], система домашней автоматизации с открытым исходным кодом, которая позволяет контролировать и настроить различные устройства, такие как огни, переключатели, датчики температуры, дождя, ветра, УФ и счетчики (электрические, газ, вода).

В рамках программы ARTEMIS исследовательский проект Arrowhead [4] и команда Smart Meter Ltd разработала и протестировала в муниципальных системах брокера интеллектуальные сервисы, подходящие для водораспределительной сети, для установки на подстанциях централизованного теплоснабжения и для измерения температуры и влажности в офисах и многоквартирных домах. В этом проекте был разработан брокер, который применяет сервисы оркестровки, на шлюзе, который предоставляет функции преобразования адаптеров и протоколов, а также на инструменте для соединения аппаратных устройств, API-интерфейсов и онлайн-сервисов [4]. Благодаря усовершенствованию датчиков и шлюзов становится возможным проектировать и внедрять общую архитектуру для коммунальных систем муниципалитета. Интеллектуальные сервисы полагаются на прямое взаимодействие между устройствами и веб-сервисами, например, для отслеживания состояния устройства в режиме реального времени из удаленных объектов, таких как центры технического обслуживания и ремонта.

В проекте Arrowhead мы исследовали способы, как взаимодействовать и контактировать с устройствами в нескольких слоях и иерархиях сетей от облачной службы через Интернет, GSM или другие сети напрямую до устройств. Возможна маршрутизация через клиентскую SCADA или другие системы автоматизации, которые используют устаревшие технологии.

Целью данной работы является предоставление практической реализации сенсорной системы для управления инфраструктурой устаревших и интеллектуальных муниципальных систем, которая соответствует подходу инфраструктуры промышленной автоматизации. Поэтому в этой статье мы обсуждаем подход к объединению интеллектуальных муниципальных систем, основанный на COA, когда обмен данными датчиков осуществляется через сер-

висы. Предлагаемая технология основана на Arrowhead Framework [11] и решение было основано на сервисах, предоставляемых брокером сервисов на основе MQTT. Предложенные брокером интеллектуальные сервисы применяет подход визуального программирования потока данных [16]. Node-RED [9] был выбран среди других инструментов, чтобы доказать жизнеспособность и преимущества предлагаемой архитектуры, реализованной в качестве локального облака автоматизации систем Smart City. Брокер служб находится на шлюзе, который обеспечивает функции преобразования адаптеров и протоколов, а также на инструменте для соединения аппаратных устройств, API-интерфейсах для онлайн-сервисов.

Кроме того, мы хотели бы продемонстрировать, что разработанный подход применим и в других областях, находящихся за пределами автоматизированного управления инфраструктурой муниципальных систем. Для этого мы начали применять брокер MQTT в разных сферах, в частности для развития автономной системы пчеловодства. Мы описываем настройку 1-го этапа этой системы в главе «Сервисный брокер на основе MQTT для автономной системы пчеловодства».

Сенсорная система для управления инфраструктурой устаревших и интеллектуальных муниципальных систем. СОА на основе публикации и подписки на очередь сообщений (MQTT)

В качестве альтернативы реализации веб-сервиса была разработана архитектура СОА, основанная на очереди сообщений с функцией публикации и подписки (рис. 1).

Для его реализации используется протокол Транспорт Телеметрии в Очереди Сообщений (MQTT), стандартизированный в OASIS [5]. Это технология соединения «машина к машине» (M2M) в «Интернете вещей» обеспечивает упрощенную передачу сообщений публикации/подписки со следующими преимуществами [6], [7]:

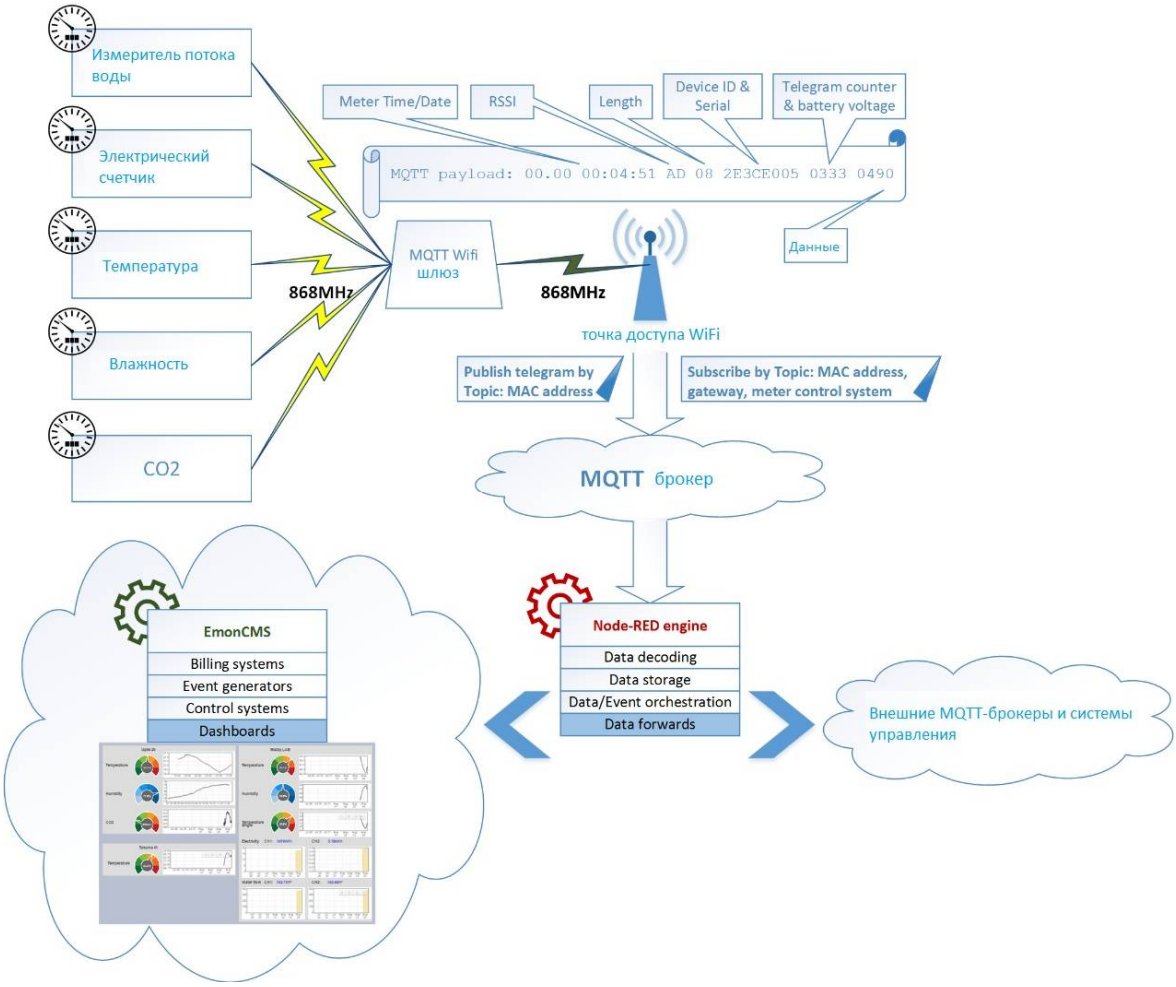


Рис. 1. Сервисный брокер на основе MQTT

- расширение возможности подключения за пределы предприятия к интеллектуальным устройствам;
- оптимизированные варианты подключения, для датчиков и удаленных устройств;
- предоставление соответствующих данных любому интеллектуальному активу принятия решений, который может их использовать;
- обеспечение масштабируемости развертывания и управления решениями.

Протокол MQTT основан на принципе публикации сообщений и подписки на темы. Сообщения MQTT публикуются устройством провайдера в темах в брокере сообщений (см. рис. 1). Потребитель в свою очередь подписывается на получение определенных сообщений, подписываясь на тему в брокере. Посредник сообщений берет на себя роль Сервисного реестра и системы авторизации [11]. Сервисный реестр можно рассматривать как предметную область. Подписки могут быть определенными, ограничивая сообщения, которые получены к конкретной теме, или они могут использовать подстановочные знаки, такие как знак числа (#), для получения сообщений по различным связанным темам [6]. По причине безопасности шифрование по сети будет осуществляться с использованием SSL/TLS независимо от самого протокола MQTT. Система оркестратор будет такой же, как для решения веб-службы: механизм бизнес-процессов на основе BPMN.

Наиболее важным преимуществом этого решения по сравнению с реализацией веб-службы является ее «push концепция», которая не требует открытых входных портов на сто-

роне владельца устройства. Кроме того, это позволяет сравнительно просто реализовать решение на основе проверенных методов и существующих инструментов.

Мультиисточниковые датчики и шлюзы

Внедренная система состоит из набора специально разработанных модулей – «умных» измерителей, обеспечивающих различные типы данных телеметрии и приложений управления:

- измерения звука - выборочные усредненные мониторинг шума окружающей среды;
- измерения давления - системы водоснабжения;
- потока воды – расход воды для бытовых счетчиков;
- счетчики электроэнергии - импульсный интерфейс для общих счетчиков электроэнергии;
- измерители температуры;
- измерители влажности;
- счетчики CO2;
- тензометры - используются для регистрации деформации и масштабирования.

Такой список интеллектуальных счетчиков может представить большинство систем коммунального контроля, работающие на территории города или муниципалитета.

Предлагаемое решение для связи как часть системы коммунального обслуживания содержит многоинтерфейсную модульную платформу с двумя компонентами основного узла. Узел измерения - подключается к измерителю через переключаемые/выбираемые интерфейсы (токовая петля, оптический интерфейс IEC1107 и т.д.). Межсистемная связь возможна с использованием выбираемых интерфейсных модулей (например, IEEE 802.3, радио интерфейс ISM, GSM / GPRS). Узел шлюза - имеет выбираемую архитектуру межсистемного и внутреннего интерфейса связи. Узел шлюза обеспечивает запросы, предварительную обработку показаний, безопасную доставку данных, постановку в очередь и т.д. Узлы шлюза состоят из радиомодуля и модуля Wi-Fi 802.11b / g / n, в который встроен микроконтроллер (ESP8266). Модуль Wi-Fi поддерживает режимы как станции (режим клиента Wi-Fi), так и режима точки доступа. Узлы шлюза в качестве опции имеют модуль для связи GSM/GPRS.

Веб-интерфейс отображает информацию о шлюзе и позволяет выполнять настройки вручную клиента Wi-Fi. Также может быть инициировано автоматическое сканирование SSID для сканирования доступных сетей Wi-Fi, а затем веб-интерфейс позволяет вводить данные и параметры авторизации в виде *hostname/ip* адреса брокера MQTT и данных авторизации для продолжения передачи телеграммы устройства учета с использованием Wi-Fi. *connection.ed* пароля. Шлюз автоматически перенаправляет подключенное клиентское устройство на веб-интерфейс конфигурации.

Сервис оркестровки

Оркестровка в контексте СОА может рассматриваться как система, которая поддерживает создание всех других систем посредством предоставления услуг координации, управления и развертывания. Система ***Оркестратор*** является центральным компонентом Arrowhead Framework, а также в любой архитектуре на основе СОА [8]. Оркестровка используется для управления как системы развертываются и каким образом они должны быть взаимосвязаны.

Мы развернули Node-RED [9] в качестве инструмента оркестровки для соединения аппаратных устройств, API-интерфейсов и онлайн-сервисов новыми и интересными способами. В основе Node-RED лежит визуальный редактор, позволяющий связывать сложные потоки данных специалисту с небольшими навыками программирования. Основная функция

Node-RED заключается в декодировании и маршрутизации данных интеллектуального измерения MQTT для дальнейшей оркестрации или использования во внешних службах в качестве систем биллинга или мониторинга клиентов (см. рис. 2).

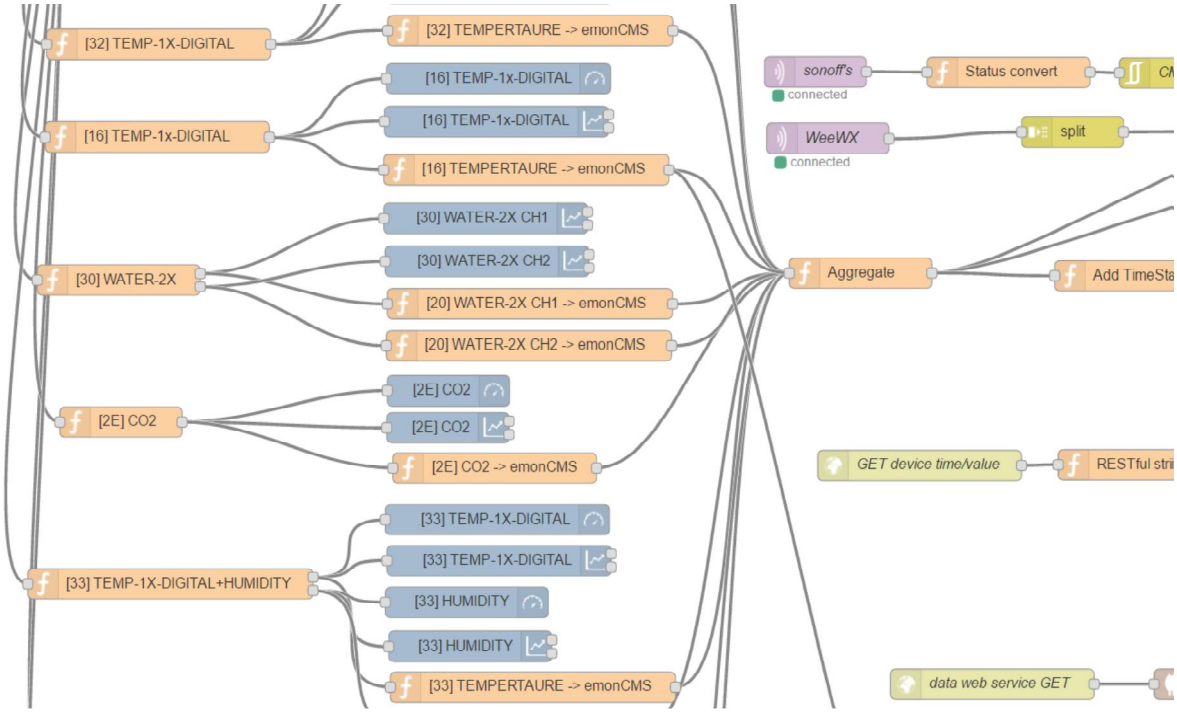


Рис. 2. Внедрение сервисов брокера с использованием потоков Node-RED на основе MQTT сервисов

Для декодирования необходимо выполнить операцию формирования полезной нагрузки и направить ее в службу хранения и визуализации данных с использованием подхода IoT:

- Первый шаг - определить источник MQTT, который является шлюзом. Шлюзы публикуют полученные данные измерений в теме на основе своего MAC-адреса, который также служит в качестве службы конфигурации путем подписки на MAC / конфигурацию, соответствующие уровни QoS могут быть указаны как для опубликованных, так и для подписанных тем.
- Следующий узел - «SERIAL + VALUE» - это создает массив элементов из начальной полезной нагрузки (телеграмма), каждый элемент массива обрабатывается отдельно в зависимости от необходимого вывода или последующей обработки.
- Новый объект создается путем деления имени/пары значений, где важными элементами являются: серийный номер - который определяет тип данных и формат кодирования, а также уникальный идентификатор устройства, который отличает измерительные устройства. Второе значение - это блок данных, необходимый для процедур декодирования.
- Двойной вывод состоит из двух новых сообщений MQTT, содержащих отдельные данные от каждого датчика, с темой типа измерения и уникальным серийным номером устройства.

Для дальнейшей обработки данных и мониторинга используется кодовая база Emoncms [12]. Серийные вызовы API JSON подготавливаются с использованием отдельного узла «(33) TEMPERTAURE -> Emoncms» и «(33) HUMIDITY -> Emoncms». Привязка данных к Emoncms осуществляется с использованием общего ключа API. Для упрощения идентифи-

кации различных типов датчиков номер типа включен в описание узлов в скобках (пример. Узел (33) представляет собой один цифровой датчик температуры с датчиком влажности).

Рабочий прототип сервис-брокера на основе MQTT для оркестрации сервисов находится и поддерживается в облаке серверов «Вентспилсский цифровой центр» (ВЦЦ).

ВЦЦ принадлежит Вентспилсской городской думе как учреждению, отвечающему за развитие и поддержание инфраструктуры ИКТ в городе Вентспилс, Латвия. МСП «Smart Meter» - партнер Рижского технического университета по проекту Arrowhead - отвечает за установку и обслуживание датчиков на объектах и в помещениях клиента.

Вывод

Поскольку коммунальные службы в муниципалитетах в настоящее время занимаются автоматизацией процессов, но устанавливают и обслуживают очень разветвленные системы, вопрос оптимизации затрат на обслуживание становится очень важным, особенно когда затраты на обслуживание вызывают рост тарифов на коммунальные услуги. Применение подхода Arrowhead Framework для оптимизации коммунальных систем муниципалитета демонстрирует многообещающие возможности для экономии затрат на обслуживание систем. Девять клиентов Maintenon, представляющих небольшие сервисные компании по недвижимости и частных домработниц, используют систему управления, основанную на сервисном брокере. Приблизительно триста расходящихся интеллектуальных счетчиков, шлюзов и приводов были установлены в нескольких городах Латвии. В определенном подходе сервисный брокер демонстрирует способы, как взаимодействовать и контактировать с устройствами через несколько уровней и иерархий сетей, в облаке сервисных услуг через Интернет, GSM или другие сети. Мы также продемонстрировали, что брокер услуг MQTT можно использовать для автоматизации процессов в других сферах, например, в качестве центральной части автономной системы пчеловодства.

Развертывание Node-RED в качестве инструмента распределенного потока данных для подключения датчиков, узлов и шлюзов обеспечивает экономию времени и средств по сравнению с традиционными методами программирования. Однако нам необходимо создать надежные индикаторы, нацеленные на оценку производительности сетевых подключений и затрат на настройку и обслуживание таких систем мониторинга и управления.

Библиографический список

1. Bassi, A., Bauer, M., Fiedler, M., Kramp,T., van Kranenburg,R., Lange, S., Meissner, S., 2013. Enabling Things to Talk – Designing IoT solutions with the IoT Architectural Reference Model. Springer

2. Blomstedt, F., Ferreira, L.L., Klisics, M., Chrysoulas, C., de Soria, I.M., Morin, B., Zabasta, A., Eliasson, J., Johansson, M., Varga, P.: The arrowhead approach for SOA application development and documentation. In: IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society. pp. 2631-2637 (Oct 2014).

3. Varga, P., Blomstedt, F., Lino Ferreira, L., Eliasson, J., Johansson, M., Delsing, J., Martinez de Soria, I.: Making system of systems interoperable - the core components of the arrowhead technology framework. Journal of Network and Computer Applications (2016).

4. Arrowhead framework wiki. [Online]. Доступно по: https://forge.soa4d.org/ plugins/media wiki/wiki/arrowhead-f/index.php/Main Page [accessed September 2017].

5. “OASIS Message Queuing Telemetry Transport (MQTT) Technical Committee | Charter.” [Online]. Доступно по: https://www.oasisopen. org/committees/mqtt/charter.php. [Accessed: October 2017].

6. V. Lampkin, W. T. Leong, L. Olivera, S. Rawat, N. Subrahmanyam, and R. Xiang, Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry. IBM, 2012.

7. “MQ Telemetry Transport.” Доступно по: http://mqtt.org/. [Accessed: October 2017].

8. P. Varga, C. Hegedus, Service Interaction through Gateways for Inter-Cloud Collaboration within the Arrowhead Framework Conference: 5th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronics Systems (Wireless VITAE), At Hyderabad, India https://www.researchgate.net/publication/287878312.

9. Open Energy Monitor project, Node-RED, http://guide.openenergymonitor.org/integrations/nodered/ [Accessed: October 2017].

10. Open Energy Monitor project, Web-app for processing, logging and visualizing energy, temperature and other environmental data https://emoncms.org/. [Accessed: October 2017].
11. J. Delsing, ed., IoT based Automation - made possible by Arrowhead Framework. CRC Press, Taylor & Francis Group, 2016.
12. Open Energy Monitor project, Web-app for processing, logging and visualizing energy, temperature and other environmental data https://emoncms.org/. [Accessed: October 2017].

УДК 004.065; ГРНТИ 20.15.05

О СОСТОЯНИИ ГОСУДАРСТВЕННЫХ РЕГИОНАЛЬНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ В РЯЗАНСКОЙ ОБЛАСТИ

Е.А. Кутузов
Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, eakutuzov@mail.ru

Аннотация. В работе рассматривается состояние процессов по построению единой системы управления государственными информационными системами и сервисами в Рязанской области.
Ключевые слова: информационные системы, цифровое правительство, региональное управление.

ABOUT THE STATE OF STATE REGIONAL INFORMATION SYSTEMS IN THE RYAZAN REGION

E.A. Kutuzov
Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, eakutuzov@mail.ru

Abstract. The paper considers the state of the processes for building a unified management system for state information systems and services in the Ryazan region.
Keywords: information systems, digital government, regional management.

В настоящее время, Стратегией развития Рязанской области до 2030 года определены амбициозные цели, являющиеся важными для граждан [2]. Поставлена задача создать опорную инфраструктуру, для обеспечения быстрого и эффективного перехода к цифровизации всех сфер. Одной из региональных программ, которая призвана значительно повысить эффективность работы региональных учреждений, межведомственного взаимодействия, качество и оперативность принятия решений, является программа «Развитие информационного общества…», реализуемая с 2014 года и претерпевшая неоднократные изменения [1,3]. Среди программных мероприятий запланировано развитие существующих информационных систем, приобретение программного обеспечения, сопровождение программных продуктов и др. К сожалению, все эти меры не позволяют достигнуть анонсированные руководством региона планы по построению платформы управления регионом, включающую цифровое правительство, единую систему бухучета и ряда других направлений [5].

Как раз на министерство цифрового развития информационных технологий и связи Рязанской области возложены полномочия по координации мероприятий использования информационно-коммуникационных технологий, созданию, развитию, модернизации, эксплуатации информационных систем и информационной инфраструктуры в деятельности органов государственной власти региона (далее – РОИВ) и подведомственных им учреждений. Однако в рамках политики информатизации, по-прежнему не сформирована дорожная карта по построению единой системы управления государственными информационными системами и сервисами в Рязанской области. На практике это отражается в том, что каждый орган власти в рамках своего бюджета самостоятельно осуществляет мероприятия по закупке и внедрению информационных систем для подведомственной сферы.

РОИВами используется 57 программных продуктов разных разработчиков [4], основанных на различных платформах и интерфейсах. Это приводит к ряду проблем:

- несопоставимость и трудность (зачастую невозможность) интеграции различных программ, необходимость сотрудников работать в каждой из программ отдельно, дублировать информацию;
- отсутствие единой информационной среды Рязанской области, баз данных и знаний, сведений и информации, отсутствие системности, что снижает достоверность информации и усложняет ее поиск;
- «ручной» ввод информации в периодические отчеты, невозможность формализации и автоматизации работы;
- разовое формирование информации по конкретным запросам
- отсутствие системы контроля внедрения и использования информационных систем, низкая степень использования функционала.

Таким образом, для реального повышения эффективности работы государственных информационных систем Рязанской области необходима централизация и формирование единой платформы с перспективным подключением всех органов государственной власти Рязанской области, органов власти местного самоуправления Рязанской области и подведомственных им учреждений (рис.1).

Рис. 1. Единая платформа регионального управления

Единая платформа должна быть интегрирована с шестью сервисами:

1) облачное решение;
2) сертифицированная система безопасности;
3) единый центр обработки данных (ЦОД);
4) поддержка законодательных изменений;
5) единая служба поддержки;
6) мобильное приложение.

Организация документооборота на единой платформе позволит структурировать и повысить эффективность финансово-хозяйственной деятельности 1500 учреждений региона и 18 000 сотрудников. Реализация бюджета, осуществление закупок, исполнение заданий станут более прозрачными и легко управляемыми благодаря системам управления и автоматизации отношений:

1) корпоративный портал (Битрикс 24);

2) реестр Имущества;
3) реестр Кадров;
4) управление Программами.

На базе единой платформы всех данных формируется массив аналитических данных, по следующим направлениям:

1) динамическая Стратегия 2030;
2) гражданские и контрольные сервисы;
3) ситуационный Центр Губернатора;
4) управление мотивацией – KPI.

Преимущества от создания единой платформы:

- единое техническое исполнение;
- целостность и однородность;
- один Центр обработки данных (ЦОД);
- региональный центр компетенций;
- 1500 готовых специалистов в Рязани;
- своя система обучения;
- снижение коррупционной составляющей.

Единая платформа так же предполагает интеграцию учреждений здравоохранения, образования, службы (Занятости, Ростехнадзора, Роспотребнадзора и др.), культуры (музеи, театры, библиотеки).

Сервисы для образовательных организаций автоматизируют процессы для всех участников системы образования. Граждане (родители и обучающиеся) получат доступ к электронным и мобильным сервисам, смогут заказывать госуслуги. Органы управления образованием (региональные и муниципальные) смогут осуществлять учет контингента и педагогических кадров, управление подведомственными организациями, проводить мониторинг и оценку качества образования, интегрировать региональную систему с федеральными (ЕПГУ, федеральный сегмент ГИС «Контингент» и т.д.) оказывать госуслуги по записи в образовательные организации. Образовательные организации региона автоматизируют управление административно-хозяйственной деятельностью, учет кадров и контингента (электронные портфолио), учет платных образовательных услуг и стипендий, расписание и учет аудиторного фонда, переведут документооборот, журналы и дневники в электронный формат.

Сервисы для медицинских организаций будут функционировать по трем блокам:

1) Медицинская информационная система, включающая администрирование потоков пациентов, биллинг, ЭМК;

2) Информационная система для вспомогательных служб и параклиники, включающая больничные аптеки, диетическое питание, клинические лаборатории;

3) Финансово-хозяйственная система, включающая бухгалтерию учреждения, зарплату и кадры учреждения. На выходе каждого блока формируется информация об объемах медицинской помощи, о прямых затратах и о затратах в целом. По результатам всех полученных сведений производится расчет себестоимости медицинской помощи (в т.ч. по законным случаям).

Единая система управления регионом формируется из 1) оперативных и реестровых систем управления финансами и ресурсами региона и 2) облачной системы управления ФХД учреждений «ERP-Бюджет», которые вместе формируют 3) систему стратегического планирования, анализа и управления. Система стратегического планирования, анализа и управления включает: стратегическое планирование (определение KPI), анализ и мониторинг показателей, ГЕО-портал, публикация открытых данных.

Таким образом, создание «единой платформы регионального управления» в Рязанской области, позволит получать и обрабатывать разнородные массивы информации обо всех важных событиях и оперативно реагировать на их изменения, тем самым ускорив на местах

проведение новой административной реформы по широкой автоматизации деятельности органов государственной власти и органов местного самоуправления, направленной на построение системы эффективного управления.

Библиографический список

1. Постановление Правительства Рязанской области от 29.10.2014 № 307 «Об утверждении государственной программы Рязанской области «Развитие информационного общества, инновационной деятельности и промышленности».

2. Распоряжение Правительства Рязанской области от 29.10.2014 № 307 «Об утверждении Плана мероприятий по реализации Стратегии социально-экономического развития Рязанской области до 2030 года.

3. Кутузов Е.А. / Согласованность показателей региональных программ с показателями эффективности деятельности органов исполнительной власти субъектов РФ / Современные тренды развития науки: Сборник статей II Всероссийской научно-практической конференции (21 ноября 2018 г.). / Нижний Новгород: ООО «АРС-Рейтинг» / 2018 г. / 15-24 с.

4. Перечни информационных систем, банков данных, реестров и регистров, находящихся в ведении министерства цифрового развития информационных технологий и связи Рязанской области / [Электронный ресурс] Режим доступа: https://it.ryazangov.ru/department/info/. – Дата доступа: 25.01.2020.

5. Цифровой комитет создан при правительстве Рязанской области / [Электронный ресурс] Режим доступа: http://d-russia.ru/tsifrovoj-komitet-sozdan-pri-pravitelstve-ryazanskoj-oblasti.html. – Дата доступа: 25.01.2020.

УДК 621.396; ГРНТИ 47.47

WEB ПРИЛОЖЕНИЕ ДЛЯ УДАЛЕННОГО ИЗУЧЕНИЯ ЯЗЫКА JAVASCRIPT

Н.В. Всемирнов, А.Н. Пылькин

Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, vsemir4ik@mail.ru

Аннотация. В работе рассматриваются некоторые уже существующие решения, а также средства разработки web приложения.

Ключевые слова: web приложение, react, node.js, JavaScript.

WEB APPLICATION DEVELOPMENT FOR REMOTE LEARNING JAVASCRIPT

N.V. Vsemirnov, A.N. Pilkin

Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan,vsemir4ik@mail.ru

The summary. The paper discusses some existing solutions, as well as development tools for web applications.

Keywords: web application, react, node.js, JavaScript.

Язык JavaScript может использоваться как и для разработки пользовательского интерфейса и клиентской части web сервиса, так и для программно-аппаратной части сервиса. Например, Netflix, Uber и eBay используют его в своих платформах. Поэтому стоит заинтересовать новых разработчиков и дать им основные знания языка JavaScript.

Существует большое количество общедоступных web ресурсов для удаленного изучения языка JavaScript. Но большинство из них имеют существенные недостатки. В предлагаемой системе сделана попытка для устранения недостатков существующих программных средств. В учебнике https://learn.javascript.ru/ очень легко получить ответ на предлагаемые задания, что дает возможность некоторым пользователям не выполнять его. Данное средство имеет достаточно сложную навигацию, что усложняет его использование (рис. 1).

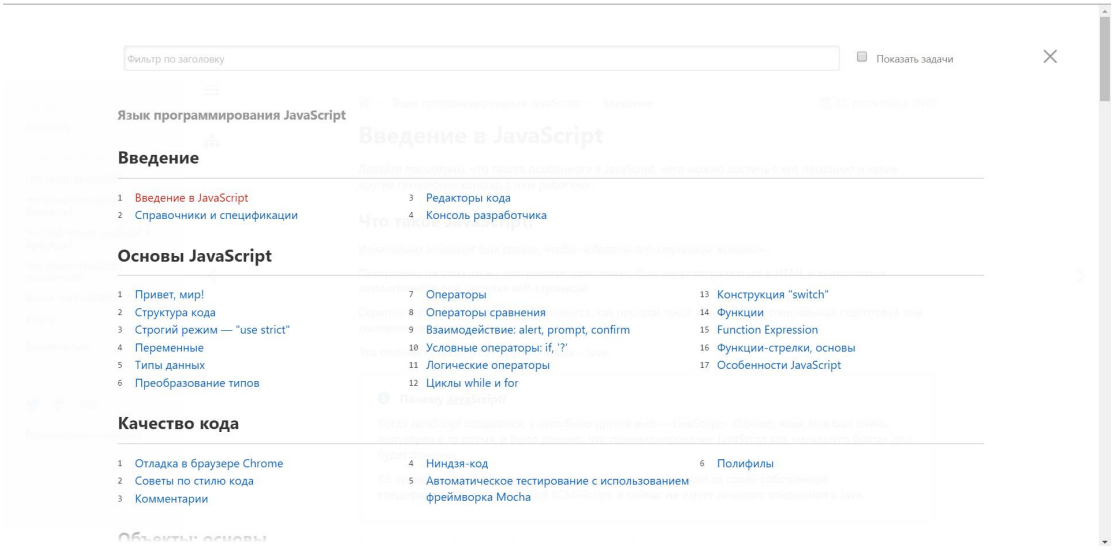


Рис. 1. Навигация на https://learn.javascript.ru/

Другие средства также имеют недостатки. Например сайт https://www.w3schools.com/js/default.asp имеет большое количество навязчивой отвлекающей рекламы (рис. 2).

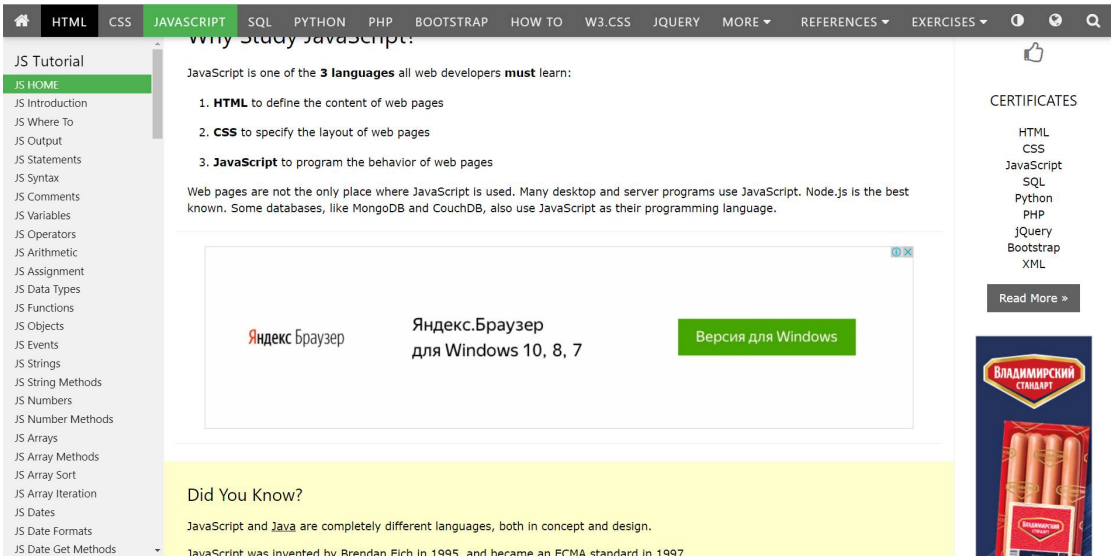


Рис. 2. Реклама на https://www.w3schools.com/js/default.asp

Разработка клиентской части

В разрабатываемом приложении целесообразно использовать фреймворки React и Node.js.

React - гибкая JavaScript библиотека для создания пользовательских интерфейсов, используя компонентный подход. Компоненты можно и нужно переиспользовать, наследовать друг от друга.[1]

Node.js — среда выполнения кода JavaScript вне браузера. Она построена на основе движка JavaScript Chrome V8. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API. API (аббревиатура от Application Programming Interface) – интерфейс программирования приложений, позволяющий сервисам взаимодействовать, получать доступ и обмениваться данными. Браузерный JavaScript использует Web

API, которые обеспечивают доступ к DOM и пользовательскому интерфейсу страниц и веб-приложений. Серверный JavaScript использует API, обеспечивающие доступ к файловой системе приложений, http-запросам, потокам.[2]

На рисунке 3 показана общая структура интерфейса, которая обеспечивает интуитивно понятный диалог в процессе обучения. Основными проблемами при реализации такой структуры являются разработка следующих частей:

1 - Навигация по разделам.
2 - Навигация по статьям.
3 - Статья.
4 - Раздел с ссылками на другие полезные ресурсы.
5 - Реклама.
6 - Header с логотипом, строкой поиска и иконкой пользователя.

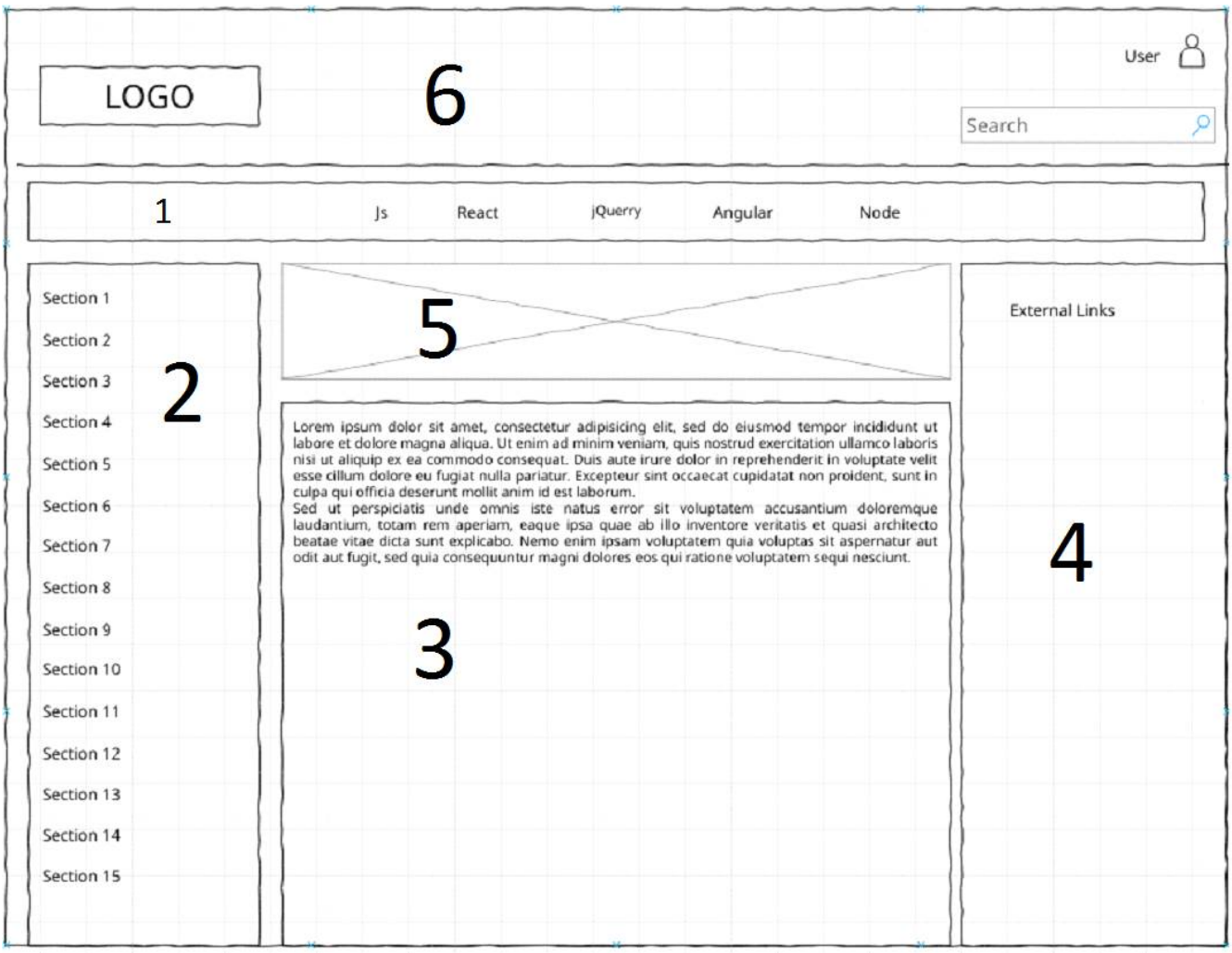


Рис. 3. Структура интерфейса

Таким образом определены средства и основные подходы для реализации предлагаемого web приложения, которое позволит удаленно изучать язык JavaScript.

1. React - JavaScript-библиотека для создания пользовательских интерфейсов URL: https://ru.react.js.org/
2 About Node.js URL: https://nodejs.org/ru/

УДК 621.311

АНАЛИЗ ВЛИЯНИЯ ЧАСТОТНОГО СПЕКТРА АУДИОСИГНАЛОВ НА ЖИВОЙ ОРГАНИЗМ НА ОСНОВЕ ВЕЙВЛЕТ ПРЕОБРАЗОВАНИЯ

П.М. Шоназаров[1], Ф.Т. Холов[2], М.Л. Мирзохасанов[3], Ш.Т. Сафаров[4]
[1]*Южно-Уральский государственный университет,*
Российская Федерация,Челябинск, shonazarov1991@gmail.com
[2]*Томский государственный университет систем управления и радиоэлектроники,*
Российская Федерация, Томск, fozil_1990-90@mail.ru
[3]*Таджикский технический университет имени академика М.С. Осими,*
Таджикистан, Душанбе, mirzo1978.78@mail.ru
[4]*Таджикский технический университет имени академика М.С.Осими,*
Таджикистан, Душанбе, shohin2015@mail.ru

Аннотация. В данной работе рассматривается частотный спектр аудиосигналов с применяем вейвлет- преобразование и влияние этих аудисигналов на живой организм. Вейвлет-преобразование (ВП) - относительно новая эффективная технология, позволяющая проводить обработку сигналов различного типа.Таким образом, он может быть эффективно применен для анализа музыкальных сигналов, как частоты в музыкальном масштабе разделены логарифмически. Обладая рядом преимуществ по сравнению с традиционными видами преобразований, ВП нашли применение в области кодирования видеоданных и изображений. Кроме того, существует множество исследований по использованию ВП для сжатия звука, в ходе которых было показано, что данный вид функций позволяет выделять различные характеристики аудиосигналов.
Ключевые слова: вейвлет-пакет, избыточные представления, частотно-временное распределение, музыка.

ANALYSIS OF THE INFLUENCE OF THE FREQUENCY SPECTRUM OF AUDIO SIGNALS ON A LIVING ORGANISM BASED ON A WAVELET TRANSFORM

P.M. Shonazarov[1], F.T. Kholov[2], M.L. Mirzohasanov[3], Sh.T. Safarov[4]
[1]*South State Ural University,*
Russian Federation, Chelyabinsk, shonazarov1991@gmail.com
[2]*Tomsk State University of Control Systems and Radioelectronics,*
Russian Federation, Tomsk, fozil_1990-90@mail.ru
[3]*Tajik Technical University named after academician M.S. Osimi,*
Tajikistan, Dushanbe, mirzo1978.78@mail.ru
[4]*Tajik Technical University named after academician M.S. Oshimi,*
Tajikistan, Dushanbe, shohin2015@mail.ru

The summary. Abstract.We consider the frequency spectrum of audio signals using wavelet transform and the effect of these audio signals on a living organism. A wavelet transforms a map more efficiently than a faster technique of fast Fourier transforms. Wavelet transform (VP) is a relatively new effective technology that allows processing various types of signals. Thus, it can be effectively used to analyze musical signals, as the frequencies on a musical scale are separated logarithmically. Having a number of advantages over traditional types of transformations, VIs have found application in the field of encoding video data and images. In addition, there are many studies on the use of VP for sound compression, during which it was shown that this type of function allows you to highlight various characteristics of audio signals.Audio data, such as speech and music, can be analyzed and processed using Fourier methods, which have, as one limitation, a constant product of time and frequency resolutions. This problem can be solved by applying the WT. Using the WT transform, you can analyze the temporal and spectral properties of non-stationary signals such as sound.
Keywords: wavelet packet, redundant performance, time-frequency distribution, music.

Цель данной статьи состоит в оценке корреляционной связи спектров известных музыкальных произведений с низкочастотными флуктуациями микроволнового излучения Солнца, достигающего поверхности Земли, - основного источника формирования и эволюции регуляторных систем организмов. На основе корреляционной обработки спектров музы-

кальных произведений известных композиторов доказывается высокий уровень их связи с низкочастотными флуктуациями микроволнового излучения Солнца, достигающего поверхности Земли. Выявленную закономерность можно интерпретировать так, что произведения известных композиторов есть не что иное как отражение в авторской обработке реальных природных процессов, к которым можно отнести флуктуации микроволнового излучения Солнца. Полученный результат может быть положен в основу обоснования необходимой процедуры определения тех или иных музыкальных произведений для их использования в лечебных целях [1].

Воздействие музыкотерапии оказывает влияние на организм человека на двух уровнях. На психологию – эта настроение и эмоции человека и на физиологию человека – эта биохимические процессы в организме человека. Используя музыкотерапию можно лечить не только душевные травмы, но и устранить многие заболевания [2]. Исследования звукового воздействия на состояние человека имеет долгую историю. Очевидно, что громкий шум может быть вызван беспокойствами, агрессией и может вызвать нежелательное поведение. Более высокие уровни сигналов звуковой вибрации влияют на систему с нарушением слуха. Одной из самых серьезных проблем, которые приносит для человека громкий звук – это проблема сна. В этой работе мы анализируем частотный спектр нескольких музыкальных произведений.

Реализация исследования

Реализация указанной идеи нашла отражение в разработке устройства, получившее название «Аппарат информационной микроволновой терапии АИМТ-1»1 (Рис. 1). Его модификация, основанная на создании с помощью нескольких излучающих антенн пространственно-распределенного поля электромагнитного излучения в диапазоне частот (4,0-4,3) ГГц, позволит при его поглощении возбудить в структуре мозга низкочастотные (частота 1-100Гц) пространственно разнесенные упругие колебания. Их возбуждение основано на «радиовибрационном» эффекте взаимодействия сложномодулированных электромагнитных излучений с биологическими структурами.

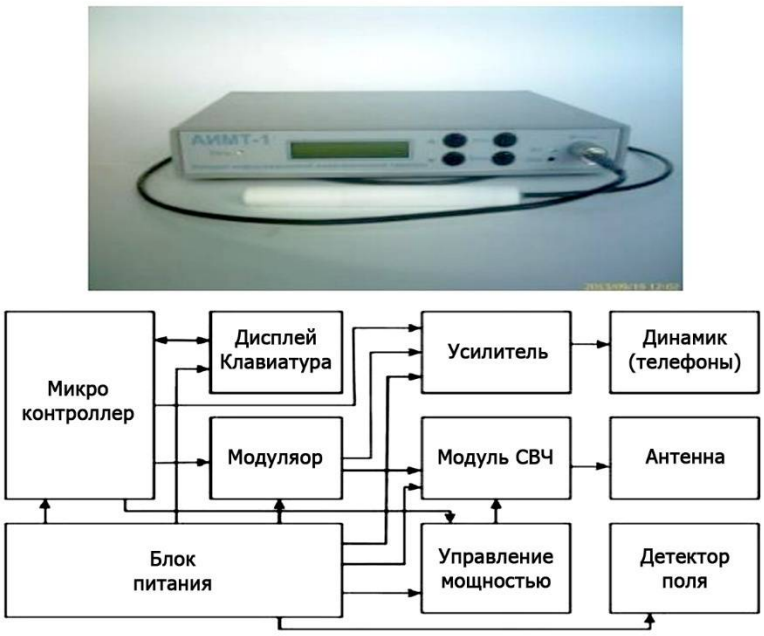


Рис. 1. Аппарат информационной микроволновой терапии АИМТ-1 и его блок-схема

Таблица 1. Характеристики прибора

Параметр	Значение
Частота	4-4,3 ГГц
Ширина полосы спектра	5-30МГц
Мощность передатчика	4-10мВт
Плотность потока мощности	50-100мкВт/см2
Вес прибора габариты	1,2кг;200х250х100мм

Данный прибор использует частотную модуляцию с несущей частотой в 4.1 ГГц и частотой девиации 26 МГц. В качестве управляющего сигнала является микроволновое излучение Солнца. В данной работе мы изменим тип управляющего сигнала на музыкальный. Для этого необходимо декодировать музыкальное произведение, используя средства программного пакета Matlab и получить функцию зависимости управляющего сигнала от времени. Также необходимо найти музыкальное произведение, максимально схожее с низкочастотными вариациями.

Экспериментальная часть

Корреляции спектра музыкальных произведений известных композиторов с низкочастотными флуктуациями микроволнового излучения Солнца

В данной работе мы анализируем нескольких молитв мусульманского народа и корелируем с низкочастотными флуктуациями микроволнового излучения Солнца. Для оценки корреляционной связи музыкальных произведений с низкочастотными флуктуациями микроволнового излучения Солнца используем следующие музыкальные композиции:

Мишари Рашид — Сура Ясин;

Абдуль Басит АбдусСамад—Сура Аль Бакара;

Абдуль Басит АбдусСамад—Азан;

Имад Али Мансури— Сура Ал-Мулк.

Для получения спектра молитв будем использовать дискретное преобразование Фурье [3,5]

$$X(k) = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn}, (k = 0, \dots, N-1), \quad (1)$$

Анализ спектров, указанных произведений (рис. 1-4) свидетельствует о наличии общих закономерностей и небольших амплитудных отличиях их распределения по частотам. Для оценки управляющей роли указанных выше молитв в лечебных целях необходимо проведение сравнительной оценки с сигналами, которые обладали безусловной управляющей ролью для организмов на всех этапах их эволюции. Среди многообразия внешних факторов, которые формировали в процессе эволюции организма его гомеостаз, приоритетная роль принадлежит Солнцу [4]. Информационный характер процессов взрывного характера, происходящих на Солнце, с помощью различных видов его излучений, достигающих поверхности Земли (электромагнитное) или околоземного пространства (корпускулярное), с большой долей вероятности лежал в основе формирования в организмах механизмов нейронной и гуморальной регуляции. Эти механизмы призваны обеспечивать в организме управляемую ритмичность процессов на различных уровнях его организации. Они являются важнейшим стабилизирующим и регулирующим фактором его внутренней среды.

Рис. 2. Спектр композиции
«Мишари Рашид –Сура Ясин»

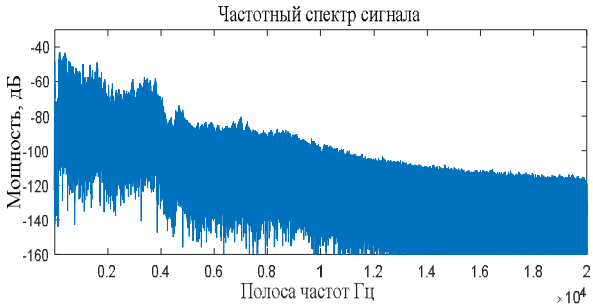


Рис. 3. Спектр композиции
«Абдуль Басит АбдусСамад - Сура Аль Бакара»

Рис. 4. Спектр композиции
«Абдуль Басит АбдусСамад – Азан»

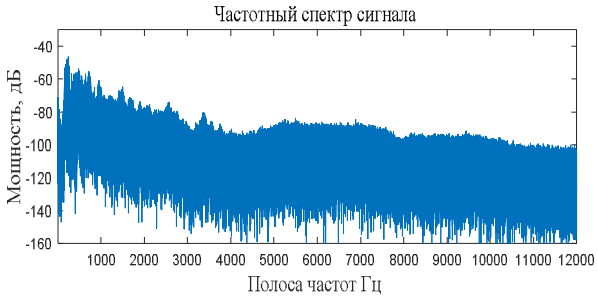


Рис. 5. Спектр композиции
«Имад Али Мансури – Сура Ал-Мулк»

Из указанных выше излучений приоритетная управляющая роль принадлежит микроволновому излучению, которое достигает поверхности Земли через 8 минут после начала излучения. Информационная составляющая этого излучения связана с его низкочастотными флуктуациями. Известна гипотеза [6], что именно эти флуктуации лежали в основе формирования в процессе эволюции организма его механизма нейронной регуляции. Сложность инструментального измерения реальных флуктуации микроволнового излучения Солнца, достигающего поверхности Земли, предопределили проведение исследований по обоснованию их модели.

Для определения «схожести» представленных выше спектров молитву y(f) со спектром модели низкочастотных флуктуаций микроволнового излучения Солнца x(f) используем процедуру расчета их коэффициента линейной корреляции [7]

$$R_{x(f),y(f)} = \frac{\Sigma\left(x(f)-1/n\,\Sigma_{f=1}^{n}x(f)\right)\left(y(f)-1/n\,\Sigma_{f=1}^{n}y(f)\right)}{\sqrt{\Sigma\left(x(f)-1/n\,\Sigma_{f=1}^{n}x(f)\right)^{2}\Sigma\left(y(f)-1/n\,\Sigma_{f=1}^{n}y(f)\right)^{2}}}. \quad (2)$$

Анализ результатов расчета коэффициентов линейной корреляции (таблица 2) указывает на высокую степень связи модели природных флуктуаций микроволнового излучения Солнца с известными моливами мусульманского народа.

Таблица 2

Название музыкального произведения	Коэффициент корреляции R_(x(f),y(f))
Мишари Рашид - Сура Ясин	0,71
Абдуль Басит АбдусСамад - Сура Аль Бакара	0,73
Абдуль Басит АбдусСамад - Азан	0,76
Имад Али Мансури - Сура Ал-Мулк	0,70

Представленные в таблице 2 результаты расчета коэффициента корреляции R_(x(f),y(f)) спектров представленных молив мусульманского народа со спектром модели природных флуктуаций микроволнового излучения Солнца следует признать неожиданными, так как они отражают неизвестную ранее высокую степень их корреляции. Исходя, из выявленной закономерности, следует, что произведения известных композиторов можно рассматривать как отражение в авторской обработке реальных природных процессов, к которым можно отнести флуктуации микроволнового излучения Солнца [8]. Полученный результат может быть положен в основу обоснования необходимой процедуры определения тех или иных музыкальных произведений для их использования в лечебных целях.

Библиографический список

1. Даровских С.Н. Микроволновая гелиобиология// С.Н. Даровских, Ю.С. Шишкова, Е.П. Попечителев, Н.В. Вдовина // Челябинск: Издательский центр ЮУрГУ, 2016. – 99с.

2. Анализ спектральных характеристик музыкальных произведений и их влияние на гомеостаз человека / П.М.Шоназаров, Ф.Т.Холов // Известия Юго-Западного государственного университета. Серия: Управление, вычислительная техника, информатика. Медицинское приборостроение. 2019. Т. 9, № 4 . С. 60–73

3. Saidov, B.B. SpectrumTransformationofanAmplitude-ModulatedSignalonanOhmicNonlinearElement / B.B. Saidov, V.I. Tambovtsev, I.I. Prokopov // ВестникЮУрГУ. Серия «Компьютерные технологии, управление, радиоэлектроника». – 2020. – Т. 20, № 1. – С. 71–78.

4. Вдовина Н.В. Устройство моделирования микроволнового излучения Солнца СВЧ диапазона для оценки его модифицирующего действия на организмы / Н.В. Вдовина, Н.Н. Гудаев, В.Н. Багаев, С.Н. Даровских, Е.П. Попечителев, Е.В. Водяницкий // Вестник ЮУрГУ. Серия: Компьютерные технологии, управление, радиоэлектроника. –2015. – Том 15. – №1. – С. 5-10.

5. Саидов, Б.Б. Преобразования спектра сигнала в активном нелинейном элементе с кубической характеристикой / Б.Б. Саидов, И.С. Следнев, В.И. Тамбовцев // Сборник трудов XXV Международной научно-технической конференции «Радиолокация, навигация, связь» (RLNC*2019). – Воронеж, 2019 – Т. 6. – С. 1–6.

6. Даровских С.Н., Разживин А.А., Кудряшова Ю.И., Кузнецов М.Е. Информационно-волновая концепция противодействия электромагнитному загрязнению окружающей среды и другим негативным факторам антропогенного происхождения. // Биомедицинская радиоэлектроника. 2008. №11. С.20–28.

7. Тихонов А.И., Самарский А.А. Уравнения математической физики. М.: Наука, 1977г., 735 стр.

8. Даровских С.Н. Радиовибрационный механизм взаимодействия биологической ткани организмов с электромагнитными полями и излучениями / С.Н. Даровских, Ю.С. Шишкова, Е.П. Попечителев, О.Б. Цейликман, Н.В. Вдовина, М.С. Лапшин // Вестник ЮУрГУ. Серия: Компьютерные технологии, управление, радиоэлектроника. – 2014. – Том 14. – №3. – С. 5-10.

УДК 681.518(075.8); ГРНТИ 28.01.75

К ВОПРОСУ УПРАВЛЕНИЯ ПРОЦЕССОМ ОБУЧЕНИЯ В УСЛОВИЯХ НЕДООСВОЕНИЯ ПРЕРЕКВИЗИТОВ УЧЕБНЫХ ПРОГРАММ

В.В. Белов*, И.Н. Филоненко**

**Рязанский государственный радиотехнический университет имени В.Ф. Уткина, Российская Федерация, Рязань, vvbeloff@yndex.ru*

*** Государственное образовательное учреждение высшего образования Московской области «Государственный социально-гуманитарный университет». Российская Федерация, Коломна, filonenkoin@mail.ru*

Аннотация. В работе рассматриваются вопросы, связанные с управлением и принятием решений в процессе планирования и реализации обучения в условиях неполноты освоения пререквизитов.

Ключевые слова: практика современного образовательного процесса, методики и технологии доучивания, дифференциация обучения, коэффициент корреляции, кластеризация, кратчайшее связывающее дерево.

TO THE ISSUE OF LEARNING MANAGEMENT IN THE CONTEXT OF UNDERDEVELOPMENT OF THE PREREQUISITES OF TRAINING PROGRAMS

V.V. Belov*, I.N. Filonenko**

**Ryazan State Radio Engineering University named after V.F. Utkin, Russia, Ryazan, vvbeloff@yndex.ru*

*** State Educational Institution of Higher Education of Moscow Region «State University of Humanities and Social Studies», Russia, Kolomna, filonenkoin@mail.ru*

The summary. The paper considers issues related to management and decision-making in the planning and implementation of training in the context of incomplete development of prerequisites.

Keywords: practice of the modern educational process, teaching methods and technologies, differentiation of education, correlation coefficient, clustering, shortest connecting tree.

Реальное положение дел с освоением пререквизитов в образовательном процессе

Независимо от стремления учебных заведений иметь как можно более подготовленный контингент и независимо от требований стандартов имеет место факт: лица, поступающие в большинство учебных заведений, имеют разную сте-пень готовности к обучению, обусловленную, в том числе, разной степенью мотивированности, интеллектуальных способностей и здоровья, Различие в степени готовности к обучению объективно проявляется в различной степени освоения пререквизитов – учебных дисциплин, являющихся базой для изучения последующих дисциплин, предусмотренных учебным планом целевого учебного заведения,

В таблице приведены данные «Минимальные баллы, дающие право на участие в конкурсе в 2019 году» одного из «средних» вузов России, находящегося по рейтингу на границе первой сотни, В «последующих» вузах ситуация, очевидно, не лучше,

Таблица 1. Минимальные баллы, дающие право на участие в конкурсе в 2019 году

Технические направления подготовки и специальности			
Направления (специальности)	Вступительные испытания		
	Математика	Физика	Русский язык
01.03.02 – Прикладная математика и информатика	33	36	36
02.03.01 – Математика и компьютерные науки	36	36	36
02.03.03 – Математическое обеспечение и администрирование информацион-	36	36	36
09.03.01 – Информатика и вычислительная техника	36	36	36
09.03.02 – Информационные системы и технологии	36	36	36
09.03.03 – Прикладная информатика	36	36	36
09.03.04 – Программная инженерия	36	36	36
10.05.01 – Компьютерная безопасность	36	36	36
10.05.03 – Информационная безопасность автоматизированных систем	36	36	36
11.03.01 – Радиотехника	33	36	36
11.03.02 – Инфокоммуникационные технологии и системы связи	33	36	36
11.03.03 – Конструирование и технология электронных средств	33	36	36
11.03.04 – Электроника и наноэлектроника	33	36	36
11.05.01 – Радиоэлектронные системы и комплексы	27	36	36
12.03.01 – Приборостроение	33	36	36
12.03.04 – Биотехнические системы и технологии	33	36	36
13.03.02 – Электроэнергетика и электротехника	33	36	36
15.03.04 – Автоматизация технологических процессов и производств	33	36	36
15.03.06 – Мехатроника и робототехника	33	36	36
15.05.01 – Проектирование технологических машин и комплексов	36	36	36
18.03.01 – Химическая технология	33	36	36
27.03.01 – Стандартизация и метрология	33	36	36
27.03.02 – Управление качеством	33	36	36
27.03.04 – Управление в технических системах	33	36	36

Баллы, представленные в таблице, соответствуют троечным оценкам, В то же время, из практики образовательной деятельности вуза, данные которого представлены в таблице, известно, что поступившие студенты имеют существенно разные средние баллы – от минимального табличного до 100 баллов,

Из того факта, что некоторые студенты, поступившие в вуз, имеют недостаточную степень освоения пререквизитов, вытекает следующее положение: организация учебного процесса в общем случае должна предусматривать процесс *доучивания* обучаемых, В то же время, в практике современного образовательного процесса методики и технологии доучивания отсутствуют, Обучаемые либо интуитивно выбирают необходимый для доучивания материал, либо попросту не осваивают новый материал, а оценки не заслуженно завышаются, – чтобы не терять контингент учащихся,

Взаимосвязь освоения начальных пререквизитов с результатами обучения

Как показывает статистика, за счет отсутствия методик доучивания учащиеся вузов «обрекаются» на своеобразное «замораживание» своего ученического статуса, – как правило, принятые в вуз троечники остаются троечниками, хорошисты хорошистами, а отличники отличниками, Имеющиеся исключения подтверждают указанную закономерность, В таблице 2 приведён фрагмент данных по выпускникам того же вуза, отражающих начальный и финальный статус учащихся, – в первой строке содержатся средние значения баллов профильных предметов по результатам ЕГЭ, а во второй – среднее значение пятибалльных оценок из приложения к диплому о высшем образовании.

Таблица 2. Средний балл ЕГЭ и средний балл в приложении к диплому

№	1	2	3	4	5	6	7	8	9
ЕГЭ, x	30.08	43.53	70.95	54.52	87.59	42.18	79.73	51.27	36.39
Диплом, y	3.39	3.25	4.93	4.47	4.76	3.23	4.93	4.38	3.49
ЕГЭ, x	40.31	99.19	38.33	30.62	67.21	72.12	41.63	61.55	33.99
Диплом, y	3.30	4.63	3.42	3.18	4.83	4.50	3.13	4.29	3.41

Взаимосвязь средних результатов ЕГЭ и выпускных оценок можно охарактеризовать значением коэффициента линейной корреляции Пирсона [6], вычисляемым по формуле:

$$r_{x,y} = \frac{Cov(x.y)}{s_x s_y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(x_i - \bar{x})^2}},$$

где $\bar{x}$, $\bar{y}$ – выборочные средние.

Коэффициент корреляции, вычисленный по данным объемом n=18, представленных в таблице 2, равен $r_{x,y}$=0.86644. Достоверность коэффициента корреляции можно определить его средней ошибке, вычисляемой по формуле: $e_r = \frac{1 - r_{x,y}^2}{\sqrt{n}}$. Традиционно значение выборочного коэффициента корреляции считают достоверным, если оно не менее чем в три раза превышает значение средней ошибки. Для рассматриваемого примера:

$$e_r = \frac{1 - 0{,}86644^2}{\sqrt{18}} = \frac{1 - 0{,}75072}{4.24264} = 0{,}05876. \quad \frac{r_{x.y}}{e_r} = \frac{0{,}86644}{0{,}05876} = 14{,}75.$$

Таким образом, полученное значение коэффициента корреляции можно считать абсолютно достоверным.

Значение $r_{x,y}$=0,86644 свидетельствует о достаточно плотной связи между степенью освоения лицом, поступившим в вуз, начальных пререквизитов с конечными результатами его обучения. Одновременно с этим, значение коэффициента корреляции более, чем на 10 процентов меньше единицы, что может быть проинтерпретировано как факт достаточно заметных изменений степени мотивированности, интеллектуальных способностей и здоровья учащихся в процессе их обучения, что вполне естественно, поскольку этим изменения способствуют естественные ситуационные и возрастные изменения.

Целесообразность группирования обучаемых

Из того факта, что студенты, поступившие в вуз, имеют разную степень готовности к обучению, естественно образом, вытекает следующее положение: студентов с разной степенью готовности к обучению необходимо учить по-разному, т, е, организация учебного процесса в общем случае должна предусматривать *несколько версий процесса обучения* (учебных траекторий). Отказ от указанного положения, имеющий место во всех вузах без исключения приводит к тому, что студенты с недостаточной степенью готовности к обучению завершают обучение с недостаточной степенью освоения надлежащих знаний, навыков и умений. О том, что разные студенты в разной степени осваивают, предусмотренные процессом обучения компетенции, говорят разные оценки в приложении к диплому. При этом выпускники с троечными дипломами в подавляющем большинстве не осваивают практически ничего из того, что предусматривалось учебными планами.

Позитивным управляющим воздействием на учебный процесс, позволяющим научить студентов «приемлемому минимуму», является организация раздельного обучения. Очевидно, что «приемлемый минимум», значительно предпочтительнее «практически ничего». Конечно же, строгая персонализация учебных траекторий (тьютеринг) требует значительных затрат, не каждому доступных и в зарубежных учебных практиках. По этой причине целесообразно осуществлять формирование учебных траекторий для групп обучаемых с примерно одинаковой степенью готовности к дальнейшему обучению. С той или иной формой разделения обучаемых по способностям и/или текущим уровнем готовности согласны многие специалисты-педагоги [1 – 4]. Наиболее систематически идея дифференциации обучении изложена в [5]. Выделение групп обучаемых может быть осуществлено с помощью процедуры кластеризации.

Аналитическое исследование качества знаний обучаемых

Для наглядного представления качества знаний контингента учащихся целесообразно построить кратчайшее связывающее дерево (Shortest path tree, SPT) [7, Глава 23. Минимальные остовные деревья] в пространстве оценок по пререквизитам.

С его помощью можно визуально выявлять факты наличия/отсутствия малых/многочисленных кластеров, ярко выраженных индивидуалов, оценивать степень вариативности уровней подготовленности к обучению. Для обеспечения возможности визуализации SPT, его построение следует осуществлять в двух- либо трёхмерном пространстве с использованием в качестве координат точек пространства оценок по наиболее важным предметам.

Построение кратчайшего связывающего дерева (версия алгоритма Прима)

Абстрагируясь от семантики рассматриваемых элементов (учащиеся учебного заведения), будем рассматривать их как точки $t^{[l]}$, $l = \overline{1, m}$ в n-мерном гиперпространстве с числовыми координатами, т. е. $t^{[l]} = (t_1^{[l]}, t_2^{[l]}, \ldots, t_n^{[l]})$. В качестве метрики в рассматриваем пространстве используем евклидово расстояние: $d_{ij} = \sqrt{\sum_{k=1}^{n} \left(t_k^{[i]} - t_k^{[j]} \right)^2}$. Метод построения SPT можно описать следующим образом.

Определение структур данных

Предполагается, что тип величин, упомянутых ранее и упоминаемых в последующем, но не включенных в ниже приводимый список, очевиден.

$D = \left[d_{ij} \right]_m^m$ – квадратная симметричная с нулями на главной диагонали матрица расстояний между точками в гиперпространстве, т. е. матрица длин ребер графа.

T_{In} – множество номеров точек включенных в SPT.

T_{Out} – множество номеров точек не включенных в SPT.

R – вектор (одномерный массив) ребер SPT.

(u, v, l) – вектор (одномерный массив) – тип элементов вектора R,

u – переменная натурального типа – номер точки из множества T_{In}, для которой найдена ближайшая точка из множества T_{Out}:

v – переменная натурального типа – номер очередной точки из множества T_{Out}, включаемой в SPT;

l – переменная натурального типа – длина ребра, соединяющего точки с номерами u, v.

i, j, k, k_1, k_2 – переменные натурального типа, используемые для представления номеров точек в гиперпространстве;

p – переменная натурального типа, используемая для индексации элементов массива R.

Последовательность действий

1. Определяется начальное значение множества T_{Out}. Его образуют натуральные числа от единицы до m, выражающие номера всех рассматриваемых точек гиперпространства:

$T_{Out} = \{1, 2, \ldots, m\}$.

2. Определяется начальное значение множества T_{In}. Его образует пара наиболее близких друг к другу (в смысле выбранной метрики) точек:

$k_1 = Arg_i(\min\limits_{1 \le i \le m-1}(\min\limits_{i+1 \le j \le m}(d_{ij})))$;

$k_2 = Arg_j(\min\limits_{1 \le i \le m-}(\min\limits_{i+1 \le j \le m}(d_{ij})))$;

$T_{In} = \{k_1, k_2\}$.

При наличии нескольких таких пар выбирается любая из них, потому что точки с равными расстояниями будут включены в SPT на последующих шагах алгоритма.

3. Определяется начальное значение счетчика выделенных ребер p =1 и заполняется первый элемент одномерного массива R с трехкомпонентными элементами, используемого для представления дерева:

$R_p =< u,. v, l >$, где $u = k_1$, $v = k_2$, $l = d_{uv}$.

4. Модифицируется значение множества номеров точек, не включенных в SPT:

$T_{Out} = T_{Out} \setminus \{k_1, k_2\}$.

5. Отыскивается номер k очередной точки для включения в SPT путем поиска во множестве T_{Out} точки, наиболее близкой к точкам из множества T_{In}:

$k = Arg_k(\min\limits_{i \in T}(\min\limits_{\substack{1 \le k \le m \\ k \notin T}}(d_{ik})))$.

Одновременно определяется и номер i точки, из ранее включенных в SPT, к которой точка с номером k наиболее близка:

$i = Arg_i(\min\limits_{i \in T}(\min\limits_{\substack{1 \le k \le m \\ k \notin T}}(d_{ik})))$.

При наличии нескольких таковых точек выбираются любые из них.

6. Модифицируется значение множества номеров точек, не включенных в SPT:

$T_{Out} = T_{Out} \setminus \{k\}$.

7. Модифицируется значение множества номеров точек, включенных в SPT:
$T_{In} = T_{In} \bigcup \{k\}$.

8. Увеличивается на единицу счетчик выделенных ребер $p = p + 1$ и заполняется соответствующий элемент массива R:
$R_p = < u, v, l >$, где $u = i$, $v = k$, $l = d_{uv}$.

9. Пункты 5 – 8 повторяются до завершения построения SPT, признаком чего является значение счетчика выделенных ребер $p = m - 1$.

Вторым шагом аналитического исследования качества знаний обучаемых могут служить процедуры кластеризации, с помощью которых можно уже формально, а не визуально выявлять специфику группирования представителей контингента обучаемых и формировать группы с примерно равным уровнем готовности к обучению. В общем случае характер группирования обучаемых по их готовности к обучению может быть достаточно вариативным: во многих случаях это могут быть «пятна» – множества лиц с примерно равными интеллектуальными признаками, но нельзя исключать и редкие ситуации, такие как «кольца» и даже «вспышки».

Заключение

В данной работе изложен подход к организации управления процессом обучения в условиях недоосвоения пререквизитов учебных программ, предполагающий дифференциацию образования и построение групповых траекторий обучения. Обоснована необходимость дифференциации обучения. С использованием коэффициента линейной корреляции Пирсона показана взаимосвязь итоговых оценок по завершению обучения с оценками по пререквизитам.

В качестве формализма, позволяющего осуществить предварительный анализ текущей образовательной ситуации на основе информативного, концентрированного и наглядного представления информации, предложено кратчайшее связывающее дерево, формируемое в плоскости либо в трёхмерном пространстве по наиболее важным оценкам пререквизитов.

Разработан алгоритм построения кратчайшего связывающего дерева, представляющий собой конкретизацию по структурам хранения предметных данных алгоритма Прима.

Указано, что основой последующего аналитического исследования качества исходных знаний и механизмом группирования обучаемых могут служить процедуры многомерной кластеризации в пространстве оценок по пререквизитам.

Библиографический список

1. Гормин А.С. Психологические основы барьерной педагогики. Теория развивающего образования для одарённых подростков. – Великий Новгород, НГУ имени Ярослава Мудрого, 2003 – (18 п.л.)

2. Лейтес Н. С. Умственные способности и возраст. М.: Педагогика, 1971. – 277 с.

3. Новые педагогические и информационные технологии в системе образования: учеб. пособие для студ. высш. учеб. заведений / Е.С.Полат, М.Ю. Бухаркина, М.В.Моисеева, А. Е. Петров; под ред. Е. С. Полат. – 3-е изд., испр. и доп. – М.: Издательский центр «Академия», 2008. – 272 с.

4. Шадриков В.Д. Способности и интеллект человека. М.: Издательство современного государственного университета, 2004 – 188 с.

5. Платонова С.М. Детская одарённость. Учебно-методическое пособие. – Санкт-Петербург: Издательство Ленинградского государственного университета имени А.С. Пушкина, 2011. – 81 с . [https://studopedia.ru/18_9647_podhodi-k-ponimaniyu-detskoy-odarennosti.html; https://studopedia.ru/18_9663_differentsiatsiya-kak-sposob-resheniya-problemi-obucheniya-i-razvitiya-odarennih-detey.html]

6. Кобзарь А.И. Прикладная математическая статистика, для инженеров и научных работников. – М.: ФИЗМАТЛИТ, 2006. – 816 с.

7. Кормен, Томас Х., Лейзерсон, Чарльз И., Ривест, Рональд Л., Штайн, Клиффорд. Алгоритмы: построение и анализ, 2-е издание. : Пер. с англ. – М. Издательский дом “Вильямс”, 2011. – 1296 с. : ил.

УДК 519.865.7; ГРНТИ 06.52.00

ОРГАНИЗАЦИЯ И УПРАВЛЕНИЕ ВИРТУАЛЬНЫМИ ФИНАНСОВЫМИ ИНСТРУМЕНТАМИ В МИРОВОМ ПРОСТРАНСТВЕ

Е.Н. Евдокимова, А.А. Лукьянов
Рязанский государственный радиотехнический университет,
Россия, Рязань,e008en@mail.ru, aalukyanov@inbox.ru,

Аннотация. Важность темы организации и управления виртуальными финансовыми инструментами в мировом пространстве обусловлена тем, что в современном мире виртуальные финансовые инструменты, криптовалюты являются уже неотъемлемой частью мирового финансового пространства. Но поскольку виртуальные финансовые инструменты являются новыми финансовыми технологиями, в связи с этим важно наладить эффективное организационное и управленческое воздействие на финансовые инструменты.
Ключевые слова: криптовалюты, биткоин, майнер, ICO, блокчейн, биржа.

ORGANIZATION AND MANAGEMENT OF VIRTUAL FINANCIAL INSTRUMENTS IN THE GLOBAL SPACE

E.N. Evdokimova, A.A. Lukyanov
Ryazan State Radio Engineering University,
Russia, Ryazan, e008en@mail.ru, aalukyanov@inbox.ru,

Abstract. The importance of the topic of organizing and managing virtual financial instruments in the global space is due to the fact that in the modern world, virtual financial instruments and cryptocurrencies are already an integral part of the global financial space. However, since virtual financial instruments are new financial technologies, it is therefore important to establish effective organizational and managerial impact on financial instruments.
Keywords: crypto currency, bitcoin, regulatory legal regulation, ICO, blocking, stock market.

Понятие и сущность виртуальных финансовых инструментов

Финансовый инструмент — это договор или контракт, в результате которого у одной стороны появляется финансовый актив, а у другой финансовое обязательство или долевой инструмент. Формально финансовый инструмент это реальный (на бумаге) или виртуальный (в виде электронной записи) документ, представляющий вышеуказанный договор.

Финансы как важнейший виртуальный ресурс субъектов существует в нескольких формах. Финансы, обслуживающие реальную экономику, являются ее «отражением». Здесь существует довольно четкая взаимосвязь между ними, хорошо изученная и представленная экономическими науками, теориями и законами.

Новейшей разновидностью платежных инструментов выступают электронные платежные инструменты и виртуальные валюты [1].

Виртуальные денежные средства представляют собой средства выражения стоимости, которые находятся в обороте в цифровой форме и которые могут выступать средством обмена или хранения стоимости, но не имеет статуса законного платежного средства. Виртуальные валюты имеют отличие от электронных денег, поскольку последние выражают компьютерное обращение реальных денег, тогда как виртуальные валюты создают совершенно новый финансовый инструмент [2].

Виртуальные денежные средства могут быть администрируемыми и неадминистрируемыми. Первый тип – это виртуальные валюты, эмитируемые администратором, который контролирует всю систему, вводит правила ее использования, осуществляет ведение и хранение реестра транзакций, изымает валюту из обращения. К такому типу валют относятся такие как E-Gold, PerfectMoney, LibertyReserve. Неадминистрируемые виртуальные денежные средства не имеют подобной третьей нейтральной стороны, в их обращении нет единого администратора и центрального контроля. Такие виртуальные валюты принято называть криптовалютами, например, Bitcoin, LiteCoin, Etherium, Ripple.

В настоящее время наиболее популярной валютой является Bitcoin. Отсутствие единого эмитента, возможность осуществлять микроплатежи (допустимо дробление одной единицы до $1/10^8$), режим работы 24/7, отсутствие посредников в виде кредитных организаций, низкая себестоимость трансакций и отсутствие юрисдикционных границ сделало эту криптовалюту привлекательным денежным инструментом.

Система функционирования криптовалют находится в распоряжении значительного числа людей, а сама виртуальная валюта представляет собой особый шифр, вычисляемый по определенному алгоритму. Определение такого шифра реализуется группой вычислительных мощностей, всеми частными лицами, которые имеют возможность осуществления инвестирования компьютерного времени и электрической энергии в процесс производства виртуальной валюты.

Таким образом, возникает ситуация того, что значительное число финансовых операций проводится при помощи различных виртуальных валют, но в единую расчетно-платежную систему они не встроены. В итоге возникает сложная экономическая и правовая ситуация неустойчивого положения данных валют, а также прав их владельцев. Отсутствие государственного контроля создает ситуацию осуществления небезопасных транзакций, которые могут привести (и приводят) к ситуациям взлома шифров и воровства. Но, между тем, криптовалюты защищены от подделок, поскольку при производстве данной виртуальной валюты применяются криптографические способы и усиление мощности всей компьютерной сети.

Еще одной крайне спорной особенностью виртуальных денежных средств выступает анонимность их применения и расчетов с помощью них. В этом отношении виртуальные валюты имеют сходство с наличными денежными средствами. В результате вполне обосновано предположить активное и распространенное применение виртуальных валют на так называемом «черном Интернет-рынке». Уже сегодня известны случаи расследования преступлений, связанных с нелегальной продажей и покупкой наркотиков, краденных вещей и оружия при помощи криптовалюты Bitcoin [3].

Кроме того, неконтролируемые трансакции при помощи виртуальных валют способствуют развитию и расширению переводов для финансирования террористической деятельности, а также ухода от налогообложения, что, естественно, имеет негативные последствия для системы налогообложения и пополнения доходов бюджета.

На рынке криптовалют существуют и участники финансовых рынков, к ним следует отнести

- биржу виртуальных валют, которая обеспечивает возможность перевода виртуальной валюты в национальную валюту за определенное вознаграждение;
- администратор в случае использования администрируемых виртуальных денежных средств;
- пользователь – лицо, которое приобретает виртуальную валюту;
- майнер – лицо, которое при помощи применения особого программного обеспечения для разрешения сложных алгоритмов распределения производит виртуальную валюту.

В награду за свою деятельности эти участники рынка получают новые единицы виртуальных денег. Но обычные компьютеры являются не подходящими для производства, например, такой валюты как Bitcoin, для этого производства необходимы сверхпроизводительные компьютеры, объединенные в единую систему.

Организация функционирования виртуальных финансовых инструментов

Первый обмен биткоинов на реальный товар произошёл 21 мая 2010 года. Исходя из данного обмена, можно сделать вывод о том, что для 10000 биткоинов была характерна цена в 41 доллар или 1303,8 рублей (относительно курса доллара в 2009 году).

Через 2 месяца 10 тысяч биткоинов стоили уже 600 долларов (19200 рублей), к концу года – 2400 долларов (72000 рублей). К середине 2013 года их стоимость превысила миллион долларов. В марте 2017 года цена одного биткоина впервые превысила цену унции золота. В 2011 году появилась биржа MtGox, ставшая первой торговой площадкой, на которой можно было обменять биткоин на обычные деньги. По состоянию на август 2013 года около 47 % всех операций в сети Биткоин производилось через эту биржу.

Стоит отметить, что биткоин не подкреплен никакими материальными ценностями в отличие от обычных денежных средств, а также ни банки, ни эмиссионный фонд не занимаются укреплением курса криптовалюты. Это исходит из того, что цифровая валюта с момента ее формирования не имеет никакого подкрепления. Стоимость криптовалюты зависит от реального спроса и предложения на биткоин. Суть данной криптовалюты заключается в том, что ее стоимость в дальнейшем будет увеличиваться. Это объясняется тем, что затраты на формирование каждой новой цифровой монеты, превышают стоимость предыдущей.

В период с 2007 года по 2009 год был создан уникальный алгоритм биткоина. Количество майнеров было небольшим, так как биткоин был еще совсем неизвестным. Добыча его шла медленно, криптовалютой никто не управлял, таже не было четкого представления, для чего она нужна. Стоимость была невысокой - около 0,1$ за монету. В 2010 году одна из бирж предоставила майнерам возможность обменивать биткоина на реальные деньги. Трейдеры привели биткоин к всемирному уровню популярности, за этим последовал взлет стоимости BC в 2013 году до 1100$ за монету. В 2015 году биткоин упал до 200$ за монету, но за этим последовал резкий взлет курса данной криптовалюты. По состоянию на ноябрь 2017 года его уровня более чем 7000 $ за один BitCoin [4].

Курс цифровых криптовалют зависит от сложности процесса их майнинга и от спроса на цифровой актив для проведения сделок. Операции, сформированные в 2008 году биткоином возникли после 2011 года, в момент образования цифровых кошельков и формы обращения виртуальных валют между пользователями. Эмиссию цифровой валюты может произвести любой системный администратор. В настоящий момент сформировано и размещено на цифровом рынке более 1 000 виртуальных валют с разным количеством криптомонет и различным курсом приобретения.

Основной цифровой валютой является криптовалюта bitcoin. В расчетной системе участвует 21 миллион биткоинов. Курс биткоина по состоянию на 19 февраля 2020 года составляет 10 226 долларов США. Его капитализация составила более 184 миллиарда долларов США. На криптобиржевых платформах все цифровые криптовалюты котируются относительно основной криптовалюты (биткоин). Самыми популярными виртуальными валютами являются: Ethereum, Ripple, BinanceCoin, Litecoin, Monero, EOS, TRON. В конце 2015 года Ethereum провела эмиссию объёмом в $18,5 млн. Тем не менее в 2016 году курс Ethereum рухнул с $22 до $12. Ситуацию спас хардфорк, то есть разделение криптовалюты на два направления [5].

Отличие большинства криптовалют заключается в способах их генерации, которое основывается на технологии блокчейна. Под блокчейном понимается выстроенная по определённым правилам непрерывная последовательная цепочка блоков (связный список), содержащих информацию. Таким образом, виртуальные валюты различаются алгоритмом формирования цепочки блоков, процессом осуществления майнинга и эмиссионными составляющими. В финансово-цифровом пространстве образовалось множество специальных

цифровых бирж и цена виртуальных монет формируется в соответствии со спросом на данную криптовалюту. Для этого любой пользователь устанавливает на персональный компьютер или на мобильный телефон цифровой кошелек для сбережения криптовалют. Приобретение цифровых валют совершается на криптобиржах или в обменном офисе. При проведении обмена фиатных средств на криптовалюту, необходимо указывать свой цифровой кошелек для зачисления на него виртуальной валюты. В случае оплаты за товары или услуги, приобретаемые через интернет, в электронном кошельке присутствует операция «передать», что является аналогом денежного перевода. Для этого вы указываете идентификатор электронного кошелька получателя цифровых валют, сумму и подтверждаете транзакцию. Пределы по переводу электронных валют для пользователей цифровых кошельков отсутствуют. На сегодняшний день открыты ICO-площадки для вложения необходимых средств в развитие бизнеса, которые дают возможность использовать краудфандинг для осуществления финансирования на новые проекты.

Потребность в инвестициях для стартапов исключает ранние ограничения и затруднения в получении вложений. В современной финансово-цифровой системе возникает необходимость создать криптовалютный рынок, подобный фондовым и валютным рынкам. В странах Евросоюза сформировано налогообложение криптовалютных транзакций. Криптовалютный рынок поддерживают страны восточной и юго-восточной Азии (Сингапур, Япония и Южная Корея).

Для того, чтобы заранее знать возможные колебания курса биткоина, следует ежедневно внимательно изучать новости, отслеживать изменения тренда, динамики и анализировать полученную информацию.

С каждым днем популярность биткоина возрастает. За всю историю существования биткоина был осуществлен примерно 31 миллион транзакций, в которых приняло участие более 25 миллионов адресов. Однако далеко не все эти адреса используются до сих пор.

Управление функционированием виртуальных финансовых инструментов

Япония одна из первых стран, которая признала криптовалютные транзакции, и обеспечила активное привлечение финансовых активов на японские цифровые платформы. В связи с этим, криптовалюты Bitcoin и Etherium в начале апреля 2017 года стали официально использоваться в расчетно-платежных системах Японии, после чего вырос курс биткоина до 2 300 долларов США уже в мае 2017 года. Однако, цифровые денежные средства в этой стране еще не признаны полновесными финансовыми ресурсами, а считаются электронными средствами, легализация которых приведет к непременному росту их положения в стране и соответственно повысится доверие к виртуальным валютам. При данной тенденции также происходит развитие южнокорейских цифровых платформ, таких как Bithumb, Upbit, Coinone, Coinlink и Korbit. Данная страна имеет толерантное отношение к цифровым криптовалютам, поэтому увеличился рост вложений в южнокорейские финансовые проекты. С развитием соответствующей тенденции сформировались криптобиржи, осуществляющие торговлю виртуальными валютами.

Эффективность ИТ-решений оказывает огромное влияние на деятельность цифровых криптовалютных бирж. Цифровые криптобиржи обладают разными торговыми площадками, у них отсутствует унификация, нет основных вендоров на площадках, что является отличительными признаками от условий на фондовой площадке, где присутствуют 10-20 основных вендоров, формирующих работу более 80% рынка торговых площадок.

Брокер, приобретая у вендора торговую площадку, производит подключение её к поставщикам, затем обеспечивает проведение коммерческой деятельности инвесторам и трейдерам. Брокер осуществляет свою деятельность на основании лицензии, формирует коммуникацию с партнерами и с банковскими структурами, организовывает потребителям все не-

обходимые услуги. Вендор реализовывает информационные технологии для удобного пользования клиентов, осуществляющих деятельность на торговых платформах.

На сегодняшний день в России предлагается упрочнить пять элементов легитимности криптовалюты:

1) идентификация клиента;

2) верификация на основе документов, данных и информации (проверка достоверности);

3) определение бенефициарного собственника;

4) оценка целей деловых отношений;

5) мониторинг в соответствии с профилем риска. Уже можно говорить о двух возможных формах законодательного регулирования оборота криптовалюты [6]:

1. Внесение дополнительных изменений в законодательство. В случае легализации цифровой криптовалюты, потребуется внесение дополнений в законодательство о лицензировании отдельных видов деятельности, включая платежно-расчетную систему, а также изменение в налоговое и банковское законодательство. Осуществляя соответствующую тенденцию, достаточно создать зоны контроля. К первой зоне контроля будет относится идентификация. Идентифицировать обязательно необходимо пользователей, майнеров, магазины, криптобиржи и обменные пункты. Ко второй зоне: криптобиржи и обменные офисы. Данные компании обязаны обладать лицензией на право проведения криптовалютных операций, которая должна выдаваться Центральным банком. В третью зону контроля входит обналичивание виртуальной валюты. Для соответсвующей категории зоны наблюдения потребуются предложения ФАТФ.

2. Разработка дополнительного закона «Об основах регулирования криптовалюты», в котором будет отражено: законодательное определение криптовалюты и ее основные особенности, субъекты сделок, условия регулирования, меры противодействия использованию криптовалюты в преступных целях. Для решения указанного направления возникает необходимость установления доступности сведений об операциях для контролирующих органов власти, оформлять лицензирование майнинга и осуществлять аккредитацию организаций, регулирующих процессинг и др.

В настоящий момент правовой статус криптовалюты в Российской Федерации не определен, таким обзазом, виртуальная валюта не может являться официальной валютой, а также средством платежно-расчетной системы. Государственная Дума Российской Федерации рассматривает два проекта федеральных законов, связанных с криптовалютой: проект федерального закона N 419059-7 «О цифровых финансовых активах» и N 419090-7 «Об альтернативных способах привлечения инвестиций (краудфандинге)» [7]. На основании законопроекта N 419059-7 «О цифровых финансовых активах» под криптовалютой понимается вид цифрового финансового актива, формируемый и учитываемый в определенном реестре цифровых операций пользователями данного реестра в соответствии с правилами ведения реестра виртуальныхсделок.

По данным портала интернет-платежей Payspacemagazine количество пользователей в мире, которые завели биткоин-кошельки, к 2015 году достигло 41 миллиона. Однако многие из этих кошельков все еще остаются пустыми. Компаниями Coinbase и ARK InvestResearch в начале 2017 года было проведено исследование, которое показало, что более 10 миллионов человек во всем мире имеют биткоины. Исходя из данных о числе держателей биткоинов во всем мире, целесообразно указать страны, которые добывают больше всего биткоинов. В настоящее время страной, которая майнит наибольшее количество биткоинов, признан Китай. Это обусловлено тем, что майнинг требует больших затрат электроэнергии, а в Китае стоимость электричества сравнительно ниже, чем в других странах. В этой стране созданы следующие компании по майнингу биткоинов: "F2Pool", "AntPool", "BTCC" и "BW". Считается,

что эти майнинговые пулы добывают примерно 60% всех новых биткоинов. Существует мнение, что биткоин нельзя использовать в Китае, но это государство оказалось огромным в мире майнером биткоина. Запрещено применять эту криптовалюту только банковскому сектору и их сотрудникам. Они не могут принимать участия в мероприятиях биткоин-бизнеса, сотрудничать с агентами биткоин-индустрии, применять банковскую систему. Для граждан, не являющихся сотрудниками банковской системы, не запрещено использование и торговля криптовалютой.

Заключение

Криптовалюта является цифровой (электронной) валютой, которая создается и контролируется посредством шифрования (криптография).

Разработка криптовалюты (а также появление новых форм и видов валют) побудило развиваться мировую экономику в новом направлении. Это побудило структурные подразделения мировой экономической системы разрабатывать новые технологические решения для финансовой среды.

Таким образом появились биржи криптовалюты, позволившие держателям счетов в любой точке мира осуществлять покупку, продажу и обмен одной криптовалюты на другие или на национальную валюту государств. Современные криптобиржи оснащены удобным пользовательским интерфейсом, позволяет проводить операции с цифровыми активами (в том числе и вывод), а также осуществлять торговую деятельность на условиях участников.

Прогресс проведения лигитимных цифровых операций в Российской Федерации непосредственно связан с общей легализацией криптовалюты, вопреки которой оппонируют Министерство финансов РФ и ряд других ведомств.

Тем не менее, интегрировав регулирование виртуальных криптобирж в стране, могло положительно повлиять на развитие экономической составляющей страны, а также привлечь в Россию иностранных инвесторов. Управление криптовалютным рынком дает возможность значительно обезопасить пользователей торговых площадок от всесторонних рисков, которые вероятны из-за отсутствия законного правового статуса и определенной структуры регулирования цифровыми активами. Некоторые российские компании работают на криптобиржах для привлечения инвестиционного капитала через ICO-платформы.

Сейчас присутствуют автоматы, работающие с биткоинами для проведения оплат за определенный продукт. Это вполне подтверждает существующие основания, связанные с дальнейшим развитием и укреплением криптомонет в финансово-цифровой системе, что позволит в дальнейшем легально использовать криптовалюты как расчётные средства.

Криптовалюта вероятно в будущем станет децентрализованной валютой и вытеснит фиатные деньги и банки благодаря технологиям на базе блокчейн. Это откроет новую ступень развития мировой экономики, однако о вреде или пользе на текущем этапе сказать сложно.

Библиографический список

1. Попиков А.А. КриптовалютаBitcoin как финансовый инструмент виртуальной экономики // Вопросы инновационной экономики. – 2016. – Том 6. – № 2. – С. 89-106.

2. Кочетков А.В. Характеристика свойств и классификация виртуальных валют // Финансовые исследования. – 2017. – Т.21. – N3. – С.29.

3. Взлет и падение SilkRoad [Электронный ресурс]. – Режим доступа: https://bits.media/silk-road/

4. Болотов Ю. Норвегия отказалась признать Bitcoin валютой // Slon.ru интернет-издание // [Электронный ресурс]. Режим доступа URL: http://slon.ru/appheroes/norvegiya-otkazalas-priznavat-bitcoin-valyutoy1034602.xhtml

5. Об использовании при совершении сделок "виртуальных валют", в частности, Биткойн. Пресс-релиз Центрального банка России, 27.01.14// 9 [Электронный ресурс]. Режим доступа URL: http://www.cbr.ru/press/PR.aspx?file=27012014_1825052.htm

6. Сидоренко Э.Л. Стратегии регулирования криптовалюты в российском законодательстве: прогнозы на будущее // Известия Юго-Западного государственного университета. – 2017. – Т.21. – № 2(71). – С.176-177.
7. Письмо Министерства финансов РФ от 23.04.2018 03-01-11/27036 [Электронный ресурс]. – Режим доступа: Справочно-правовая система Консультант Плюс.

УДК 681.518; ГРНТИ 20.15.05

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ИНТЕРНЕТ-РЕКРУТМЕНТА В СФЕРЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

В.В. Чистотин
Рязанский государственный радиотехнический университет имени В.Ф. Уткина,
Российская Федерация, Рязань, ValeryChistotin@yandex.ru

Аннотация. В работе описывается разработка информационной системы интернет-рекрутмента в сфере информационных технологий. Приводится взаимодействие клиентской части сайта с серверной и серверной части веб приложения с базой данных.
Ключевые слова: разработка, информационная система, интернет-рекрутмент, информационные технологии, клиентская часть, серверная часть, база данных, *React*, *Node.js*, *Express*, одностраничное приложение, *npm.*

THE DEVELOPMENT OF THE INFORMATION SYSTEM OF THE INTERNET RECRUITMENT IN THE SPHERE OF THE INFORMATION TECHNOLOGY

V.V. Chistotin
Ryazan State Radio Engineering University named after V.F. Utkin,
Russia, Ryazan, ValeryChistotin@yandex.ru

The summary. The development of the information system of the internet recruitment in the sphere of the information technology is described in the work. The interaction between the client side of the site and the server side and the interaction between server side and the database is listed in it.
Keywords: development, information system, internet recruitment, information technology, client side, server side, database, React, Node.js, Express, single page application, npm.

Информационные системы являются неотъемлемой частью жизнедеятельности людей. Они оказывают влияние на выполнение бытовых дел. Развитие технологий разработки информационных систем и увеличение сложности требований по функциональной составляющей программного обеспечения привело к усложнению процесса формирования команды, способной реализовать заявленные требования в установленные сроки с фиксированным бюджетом. Информационная система интернет-рекрутмента в сфере информационных технологий предназначена для решения этой проблемы. Пользовательский интерфейс информационной системы предоставит обзорную информацию о свободных сотрудниках компании и их основных навыках. Кроме этого, описанное программное обеспечение позволит пригласить сотрудника компании для проведения собеседования на проект в два клика. Информационная система интернет-рекрутмента в сфере информационных технологий представит собой веб-приложение. Веб-приложение получит трёхзвенную архитектуру. Трёхзвенная архитектура представляет собой архитектуру с тремя компонентами: клиентская часть, серверная часть и сервер базы данных. Клиентская часть веб-приложения будет написана на языке программирования *JavaScript* с использованием библиотеки для построения сложных пользовательских интерфейсов *React*. *React* - *JavaScript*-библиотека для создания пользовательских интерфейсов [1]. Она является надёжным средством для разработки сложных пользовательских интерфейсов, имеет ряд библиотек, способствующих увеличению скорости разработки пользовательского интерфейса. Кроме этого, библиотека имеет большую аудиторию среди разработчиков по всему миру, что позволяет находить решения сложных задач с отно-

сительной простотой. Серверная часть сайта будет написана на языке программирования *Node.js*. *Node.js* — это *JavaScript*-окружение построенное на движке *Chrome V8* [2]. Эта платформа с пакетом *Express* послужит фундаментом для построения прикладных решений серверной части проекта. База данных рассматриваемой информационной системы будет *MongoDB*. *MongoDB* – это документно-ориентированная система управления базами данных. Сайт является кроссплатформенным решением и доступен для просмотра и выполнения действия над ним с любого устройства, имеющего выход в сеть Интернет. Это является основополагающим фактором выбора средств разработки. Использование встроенных возможностей языка *JavaScript* позволит выполнить сетевые запросы на сервер. Запросы с клиентской части сайта на серверную часть сайта будут осуществляться с помощью новой характерной особенности спецификации языка *JavaScript EcmaScript 2016* – методом *fetch*. Этот метод – мощное и современное средство выполнения асинхронных запросов. При разработке информационной системы и детальной проработки требований к информационной системе будут использованы диаграммы языка *UML*. *UML* – это унифицированный язык моделирования, который предназначен для описания систем. Диаграмма развёртывания представлена на рисунке 1.

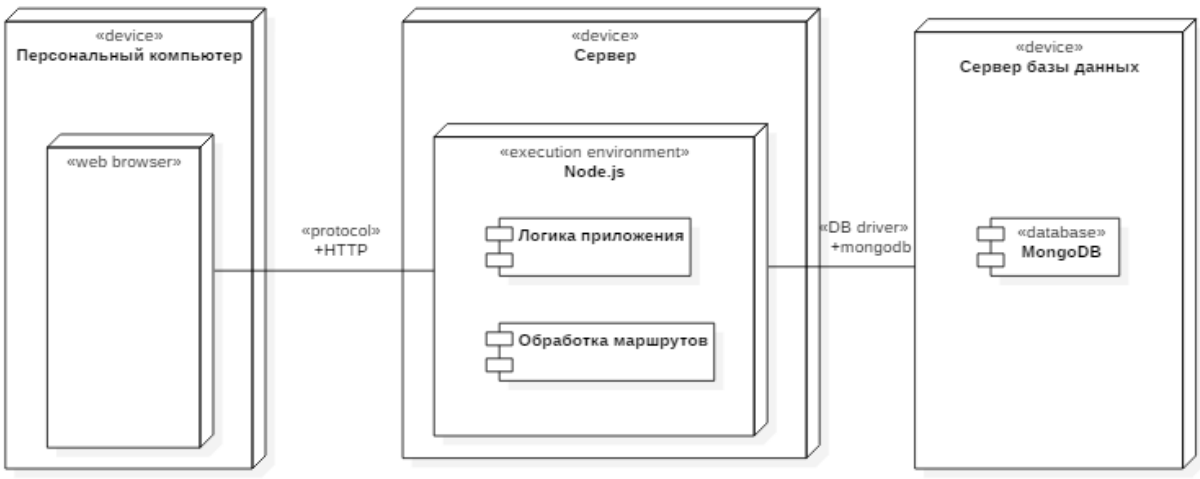


Рис. 1. Диаграмма развёртывания

Одностраничное приложение — это приложение, которое состоит из единственной HTML-страницы и прочих ресурсов (таких как *JavaScript* и *CSS*), необходимых для успешной работы. Любое взаимодействие с главной или последующими ей страницами не требует контакта с сервером, что значит — страница не требует перезагрузки [3] . В библиотеке *React* это достигается путём использования пакетов *react-router* и *react-router-dom*. Эти и другие пакеты устанавливаются пакетным менеджером *npm*.

Пользователями информационной системы будут администраторы, руководители проектов и сотрудники компании сферы информационных технологий. Администраторы будут задействованы в администрировании информационной системы. Руководители проектов являются заинтересованными лицами в найме кадров на проект. Для них доступна информация о свободных сотрудниках компании по направлениям и возможность пригласить сотрудника компании для собеседования на проект. У сотрудника компании есть возможность создать своё резюме. Сайт будет состоять из четырёх страниц. Первая страница – это страница с краткой информацией о наличии свободных сотрудников по направлениям. Вторая страница – это страница с информацией о проекте. Она заполняется руководителем проекта. Третья страница – это страница с информацией о сотруднике. На ней отображены навыки, достижения и общая информация о профессионализме сотрудника. Страница доступна для просмот-

ра руководителю проекта. Кроме этого, есть кнопка приглашения сотрудника для собеседования на проект. Четвёртая страница – это страница с информацией о сотруднике. На ней заполняются навыки, достижения и общая информация о профессионализме сотрудником. Две другие страницы – это страницы входа и регистрации. Аналогами рассматриваемой информационной системы являются сайты «Яндекс.Работа» и «*Rabota.ru*». Указанные сайты сети Интернет являются популярными системами как для найма сотрудников на работу менеджерами по управлению персоналом, так и для подачи заявлений на работу сотрудниками. Преимуществом разрабатываемой информационной системы является больший набор функциональных возможностей по сравнению с аналогами. Это обусловлено узко направленностью выбранной темы информационной системы.

Информационная система интернет-рекрутмента на проекты в сфере информационных технологий будет использована в рамках компании. Информация, которая будет использована в ней, конфиденциальна. Авторизация пользователя будет проверена на клиентской стороне сайта и на серверной стороне сайта при выполнении запроса. При определении активности не авторизованного пользователя он будет немедленно перенаправлен на страницу входа.

Регистрация на сайт доступна исключительно сотрудникам компании. Для прохождения процесса регистрации и получения полного доступа к сайту нужно подтвердить свою учётную запись и создать пароль. Письмо с просьбой о подтверждении своей учётной записи будет отправлено на электронную почту пользователя при прохождении первого этапа регистрации.

Информационная система интернет-рекрутмента на проекты является хорошим средством подбора кадрового состава на проекты в сфере информационных технологий. Эта разработка позволит получить быструю информацию о свободных сотрудниках компании по направлениям и пригласить сотрудника для собеседования на проект в два клика. Пользовательский интерфейс позволит быстро редактировать информацию о пользователе, его навыках и достижения. Информационная система интернет-рекрутмента на проекты в сфере информационных технологий является актуальным, доступным для работы на любых устройствах с выходом в сеть Интернет средством. Она подойдёт для любой компании сферы информационных технологий. Информационная система будет разработана в рамках выпускной квалификационной работы магистратуры.

Библиографический список

1. *React – JavaScript* – библиотека для создания пользовательских интерфейсов [Электронный ресурс]. – 2020. – Режим доступа: *https://ru.reactjs.org/* . Дата доступа – 22.02.2020.
2. *Node.js* [Электронный ресурс]. – 2020. – Режим доступа: *https://nodejs.org/ru/* . Дата доступа – 22.02.2020.
3. Словарь терминов *React – React* [Электронный ресурс]. – 2020. – Режим доступа: https://ru.reactjs.org/docs/glossary.html#single-page-application . Дата доступа – 22.02.2020.

УДК 519.7; ГРНТИ 28.17

МЕТОД ИДЕАЛЬНОЙ ТОЧКИ В ЗАДАЧАХ ПОСТРОЕНИЯ ТЕСТА КОНТРОЛЯ КОНЕЧНЫХ АВТОМАТОВ

Г.В. Петрухнова
Воронежский государственный технический университет,
Российская Федерация, Воронеж, gvpetrukhnova@mail.ru

Аннотация. Рассматриваются вопросы построения теста контроля конечных автоматов. Решалась задача оптимизации распределения вероятностей входных сигналов при случайном тестировании дискретных устройств..
Ключевые слова: конечный автомат, генератор псевдослучайных чисел, случайное тестирование, цифровое устройство, программный модуль, тест, энтропия распределения вероятностей.

THE IDEAL POINT METHOD IN THE TASKS OF CONSTRUCTING A CONTROL TEST OF FINITE STATE MACHINES

G.V. Petrukhnova, S.A. Prutkova
Voronezh State Technical University,
Russia, Voronezh, gvpetrukhnova@moail.ru

The summary. The questions of constructing a finite state machine control test are considered. The task was solved, relating to optimizing the probability distribution of input signals during random testing of discrete devices.
Keywords: finite state machine, pseudo-random number generator, random testing, digital device, program module, test, probability distribution entropy.

Для выделения каких-то определенных свойств объекта контроля часто используется модель, представляющую собой конечный автомат. Реализациями конечного автомата могут быть логическая схема, цифровое устройство, программный модуль либо его фрагмент, алгоритм и ряд других объектов. В рамках теории автоматов можно описать работу большого количества дискретных моделей из различных областей науки и техники. В настоящее время задачи контроля работы конечных автоматов являются актуальными. Под контролем объекта будем понимать процесс его исследования с целью проверки соответствия между наблюдаемым поведением и его истинным поведением на конечном определенным образом выбранном наборе тестовых воздействий. Тестовый контроль позволяет выявить только наличие ошибок. Диагностирование не производится, а проводится контроль по принципу «годен - не годен» (формальное тестирование).

Рассмотрим построение теста контроля с помощью генератора псевдослучайных чисел [1 - 3]. Представим конечный автомат в виде модели «серый ящик». Проверять работу реализации конечного автомата будем в контрольных точках. Контрольные точки могут содержать входы, выходы и внутренние точки схемы.

Входные воздействия и соответствующие им реакции объекта контроля представляют собой двоичные наборы заданной длины. Работу цифрового устройства можно проверить путем подачи на его входы всех возможных тестовых наборов. Если количество входов автомата велико, то тест может быть избыточным и очень длинным. Отсюда возникает задача оптимизации теста контроля.

Рассмотрим тестирование взвешенными случайными наборами, при котором на каждый первичный вход конечного автомата будут подаваться логические нули и единицы с заданной вероятностью (весом). Пусть имеется конечный автомат (КА) с L первичными входами и К контрольными точками. Пусть первичные входы конечного автомата являются независимыми и имеют различный вес (вероятность появления логической 1 на входе). На вход КА подаются случайные воздействия. Таким образом, получаем вектор весов $(u_1, \dots u_L)$ и выходную последовательность $(y_{i1}, y_{i2} \dots y_{iK})$. Представим выходную реакцию КА в виде матрицы:

$$Y=\begin{pmatrix} y_{11} & \cdots & y_{1K} \\ \vdots & \ddots & \vdots \\ y_{N1} & \cdots & y_{NK} \end{pmatrix}$$

где y_{ij} – значение, принимаемое *j*-той контролируемой точкой после подачи *i*-того входного набора;

(y_{i1} y_{i2} …y_{iK}.) – выходная реакция конечного автомата (выходной набор) на *i*-тое тестовое воздействие (*i*-тый входной набор).

Пусть качество теста внутрисхемного контроля определяется двумя критериями качества

$$H_1 = -\frac{1}{\mathrm{N}}\sum_{i=1}^{N}(w_i \cdot \ln w_i) + \frac{1}{\mathrm{N}}\sum_{i=1}^{N}((1-w_i)\cdot \ln(1-w_i)),$$

$$H_2 = -\frac{1}{\mathrm{K}}\sum_{i=1}^{K}(q_i \cdot \ln q_i) + \frac{1}{\mathrm{K}}\sum_{i=1}^{K}((1-q_i)\cdot \ln(1-q_i)),$$

где q_i – частота появления на *i*-том выходе КА (y_{1i} y_{2i} …y_{Ni}.) логической единицы;

p_k – частота *k*-того выходного набора;

w_i – частота единичного логического значения в *i*-том выходном наборе (y_{i1} y_{i2} …y_{iK}.);

N – длина псевдослучайной тестовой последовательности;

K – число контрольных точек;

$R=2^K$.

Представленные критерии являются «конфликтующими». Поэтому необходимо искать компромиссное решение, учитывающее важность каждой целевой функции, – эффективное решение.

Точку $(u_1*, \ldots, u_L*)$ называется идеальной, если в этой точке каждая из целевых функций достигает оптимального значения [4]. На практике идеальная точка достигается очень редко, но можно определить расстояние от альтернативных точек до идеальной и выбрать ту альтернативную точку, для которой расстояние будет минимальным. Метод идеальной точки позволяет исходную задачу с двумя целевыми функциями свести к решению обычной однокритериальной задачи.

Критерий H_1 достигает максимального значения при условии w_i=0.5, i=1,…,N. Критерий H_2 достигает максимального значения при условии q_i=0.5, *i*=1,…,K. Далее необходимо вычислить

$$x_j = H_j/maxH_j/$$

Затем можно определить значение целевой функции

$$R = \sqrt[2]{\lambda_1 \cdot (1-x_1^2)^2 + \lambda_2 \cdot (1-x_2^2)^2},$$

где λ_1 и λ_2 – это весовые коэффициенты, выбираемые экспериментально.

Вероятности $q_i(u), (i = 1, \dots, \text{К})$ являются неизвестными, и их получение в явном виде – сложная задача. Поэтому заменим их соответствующими частотами, вычисленными на некоторой случайной выборке достаточной длины.

Целевой критерий является случайным, и его явный вид неизвестен, поскольку выходные вероятности заменили частотами. Непосредственное вычисление значений целевого критерия и его производных невозможно. Для решения данной оптимизационной задачи используем численный метод с наблюдениями за реализациями критерия – метод покоординатного спуска. Описание метода изложено, в частности, в работе [5]. Реализация метода покоординатного спуска в рассматриваемой задаче требует оценки вероятностей появления единичного логического сигнала $q_i(u),\ (i = 1, \dots, \text{К})$ и $w_i(u),\ (i = 1, \dots, \text{N})$ на каждом выходе КА. Для этого нужно оценить вероятность появления единичного сигнала на каждом выходе КА по ее относительной частоте с достоверностью 0.95 и погрешностью не более 0.03 [1]. Таких показателей можно достичь, если использовать выборку, содержащую около 1000 наборов [1].

Исследование алгоритма оптимизации весовых множеств осуществлялось для КА, характеристики которых приведены в таблице 1.

Таблица 1. Характеристики конечных автоматов

Номер	Число входов	Число контрольных точек
1	32	8
2	32	8
3	48	8
4	60	8

В ходе экспериментов вводились одиночные ошибки в логику работы конечного автомата. Фиксировалась длина теста контроля, выявляющего внесенную ошибку, до проведения оптимизации, т.е., когда на каждый вход автомата подавалась логическая единица с частотой 0.5. Также фиксировалась длина теста контроля после решения задачи оптимизации, т.е., когда на входы автомата подавались логические единицы с частотой, отличной от 0.5. Результаты экспериментов приведены в таблице 2. Для случая 2 (таблица 2) в результате решения задачи оптимизации частоты входных единичных сигналов не изменились (остались равными 0.5).

Таблица 2. Результаты решения задач оптимизации весовых множеств

Номер	Длина теста до решения задачи оптимизации	Длина теста после решения задачи оптимизации
1	11	6
2	3	3
3	3	2
4	4	1

Экспериментальные исследования показали хорошие результаты. Предлагаемые критерии качества могут быть использованы для тестирования объектов, представляемых КА. Оптимизация множества весов для псевдослучайного теста контроля актуальна для задач контроля наличия уязвимостей в моделируемом объекте, а также для случая параллельного тестирования, позволяющего проверить, что новая (измененная) версия объекта работает так же, как и старая.

Библиографический список

1. Петрухнова Г.В. Анализ свойств симметрии бинарной матрицы в задачах тестирования цифровых устройств. – Актуальные проблемы прикладной математики, информатики и механики : сб. тр. Межд. науч. конф., Воронеж, 17–19 декабря 2018 г. – Воронеж : Издательство «Научно-исследовательские публикации», 2019. с. 1595 – 1606.
2. Пруткова С.А., Петрухнова Г.В. Тестирование конечных автоматов псевдослучайными бинарными последовательностями. – Научная опора Воронежской области: сб. тр. победителей конкурса научно-исследовательских работ студентов и аспирантов ВГТУ по приоритетным направлениям развития науки и технологий, Воронеж: ВГТУ, 2019. с. 426-428.
3. Бейзер Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем. – Санкт-Петербург: Питер, 2004.
4. Подиновский В.В., Ногин В.Д. Парето-оптимальные решения многокритериальных задач. . – Москва: Физматлит, 2007
5. Васильев Ф.П. Численные методы решения эксперементальных задач. – Москва: Наука, 1980.

УДК 62-1/-9; ГРНТИ 12.09.11

ПРОГРАММИРОВАНИЕ РЕГИСТРАЦИИ И АВТОРИЗАЦИИ ДЛЯ САЙТА

Л.А. Горохова

Рязанский государственный радиотехнический университет имени В.Ф. Уткина,
Россия, Рязань, a.g.l68@mail.ru

Аннотация. В данной работе рассказывается о написании программного кода регистрации и авторизации на языке программирования ПиЭйчПи для сайта тестирования знаний детей по основам робототехники.

Ключевые слова: сайт, регистрация, авторизация, пользователь, данные.

PROGRAMMING REGISTRATION AND AUTHORIZATION FOR THE SITE

L.A. Gorokhova

Ryazan state radio engineering University named After V. F. Utkin
Russia, Ryazan, a.g.l68@mail.ru

Abstract. This paper describes how to write a program code for registration and authorization in the PHP programming language for testing children's knowledge of the basics of robotics.

Keywords: site, registration, authorization, user, data.

Регистрация на сайте – это действия, которые направлены на то, чтобы создать на каком-либо веб-ресурсе личную учетную запись для получения возможности доступа к его опциям (расширенные возможности или доступ к страницам, просматривать которые возможно лишь после регистрации).

Таким образом, регистрация – это способ входа (или получение возможности входа) на определенный веб-ресурс. В большинстве случаев это процесс обязательный.[2]

Регистрация или авторизация пользователя является неотъемлемой частью многих веб-приложений и сайтов, и это очень важно в получении права на успех. Это начальная точка взаимодействия пользователя с вашим приложением.

ПиЭйчПи: Хайпертекст Припросессер (изначально Пёрсонал Хоум Пейдж Тулс, «Инструменты для создания персональных веб-страниц») - один из лидирующих языков современной веб-разработки. Реализация функции регистрации пользователей с помощью ПиЭйчПи является простой задачей. Логика данного процесса такова:

- вход на сайт с целью регистрации;
- ввод личных данных в специальную форму;
- проверка введенных данных сервером, создание записи о новом пользователе;
- ввод данных пользователем в форму для авторизации;
- проверка введенных данных сервером на совпадение;

- авторизация пользователя.

Это схема работы регистрации и авторизации для сайта является реализуемой. Для реализации какой-либо регистрации необходим хостинг, который поддерживает ПиЭйчПи, а также МайЭскуэль. В противном случае не удастся сохранить данные о пользователе, соответственно и зарегистрироваться он не сможет. [3]

Этот пример можно разделить на 3 части: 1) получение информации о пользователе, 2) проверка правильности предоставленной пользователем информации, 3) обработка данных для сохранения зарегистрированного пользователя в базе данных после проверки.

Третий шаг будет выполнен после того, как пользователь уже будет добавлен. Проверка уникальности данных будет выполнена на основе имени, пароля и логина пользователя, введенного им (рис. 1).

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Форма регистрации</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
    <link rel="stylesheet" href="/css/style.css">
</head>
<body>
    <div class="container mt-4">
    <!-- if (is_exist($_COOKIE['user']) )-->
    <?php

        if($_COOKIE['user'] == ''):
    ?>
    <div class="row">
        <div class="col">
            <h1>Форма регистрации</h1>
            <form action="check.php" method="post">
                <input type="text" class="form-control" name="login" id="login" placeholder="Введите логин"><br>
                <input type="text" class="form-control" name="name" id="name" placeholder="Введите имя"><br>
                <input type="password" class="form-control" name="pass" id="pass" placeholder="Введите пароль"><br>
                <button class="btn btn-success" type="submit">Зарегистрировать</button>
            </form>
        </div>
        <div class="col">
```

Рис. 1. Форма регистрации пользователя

Далее прописываются критерии ввода данных о пользователе (рис. 2).

```
<?php
$login = filter_var(trim($_POST['login']),
FILTER_SANITIZE_STRING);
$name = filter_var(trim($_POST['name']),
FILTER_SANITIZE_STRING);
$pass = filter_var(trim($_POST['pass']),
FILTER_SANITIZE_STRING);

if(mb_strlen($login) < 5 || mb_strlen($login) > 90) {
    echo "Недопустимая длина логина";
    exit();
}   else if(mb_strlen($name) < 2 || mb_strlen($name) > 50) {
    echo "Недопустимая длина имени";
    exit();
}   else if(mb_strlen($pass) < 2 || mb_strlen($pass) > 6) {
    echo "Недопустимая длина пароля (от 2 до 6 символов)";
    exit();
    }
    $pass = md5 ($pass."ghjsfkld2345");

    $mysql = new mysqli('localhost','root','root','register-bd');
    $mysql->query("INSERT INTO `users`(`login`, `pass`, `name`)
    VALUES('$login', '$pass', '$name')");

    $mysql->close();

    header ('Location: http://localhost/php/index.php');
    exit();
?>
```

Рис. 2. Ввод данных пользователя

Следующим шагом является форма авторизации и входа пользователя в веб-приложение (рис. 3).

```
        <div class="col">
        <h1>Форма авторизации</h1>
        <form action="auth.php" method="post">
    <input type="text" class="form-control" name="login"
    id="login" placeholder="Введите логин"><br>
    <input type="password" class="form-control" name="pass"
    id="pass" placeholder="Введите пароль"><br>
    <button class="btn btn-success"
    type="submit">Авторизоваться</button>
    </form>
</div>
<?php else: ?>
<p>Привет <?=$_COOKIE['user']?>. Чтобы выйти, нажмите <a href="/php/exit.php">здесь</a>.</p>
<?php endif;?>

</div>
</div>
</body>
</html>
```

Рис. 3. Форма авторизации пользователя

Введенные пользователем данные проверяются. Если информация сохранена в базе данных пользователей, то пользователь проходит авторизацию, если данные введены неверно, то выводится надпись «Такой пользователь не найден», процесс авторизации будет остановлен путем выхода. [1]

```
<?php
$login = filter_var(trim($_POST['login']),
FILTER_SANITIZE_STRING);
$pass = filter_var(trim($_POST['pass']),
FILTER_SANITIZE_STRING);

    $pass = md5 ($pass."ghjsfkld2345");

    $mysql = new mysqli('localhost','root','root','register-bd');

    $result = $mysql->query("SELECT * FROM `users` WHERE `login` =
    '$login' AND `pass` = '$pass'");
    $user = $result->fetch_assoc();
    if(count($user)== 0){
        echo "Такой пользователь не найден";
        exit();
    }
    setcookie('user',$user['name'],time() + 3600*24, "/");

    $mysql->close();

    header ('Location: http://localhost/php/index.php');

?>
```

Рис. 4. Проверка данных пользователя

В конечном итоге начальная страница будет выглядеть следующим образом (рис.5).

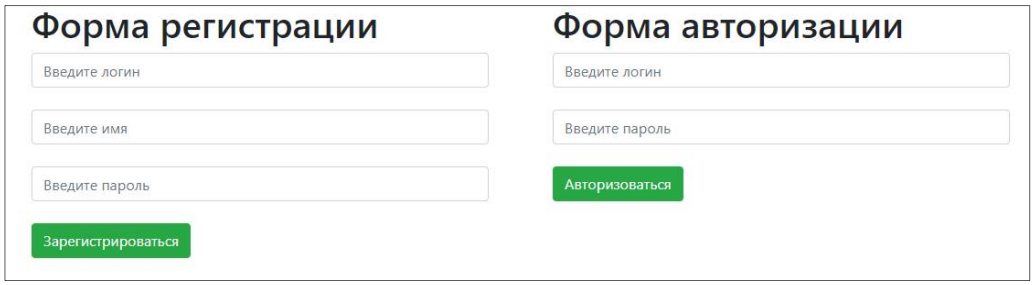


Рис. 5. Начальная страница

Следующим этапом будет настройка сайта в соответствии с потребностями и спецификой направленности. В данном случае пишется программный код на языке программирования ПиЭйчПи для сайта тестирования знаний детей по основам робототехники.

Библиографический список

1. PHP User Registration Form (Sign up) with MySQL Database. [Электронный ресурс]. - Режим доступа: https://phppot.com/php/php-user-registration-form/ - (Дата обращения: 10.02.2020)
2. Зачем нужна регистрация на web-ресурсах? [Электронный ресурс]. - Режим доступа: https://www.compgramotnost.ru/internet-gramotnost/zachem-nuzhna-registratsiya-na-web-resursah - (Дата обращения: 10.02.2020)
3. Как сделать регистрацию для своего сайта на PHP + MySQL? [Электронный ресурс]. - Режим доступа: https://you-hands.ru/2019/06/16/kak-sdelat-registraciyu-dlya-svoego-sajta-na-php-mysql/ - (Дата обращения: 10.02.2020)

УДК 005.92; ГРНТИ 20.53.17

СОЗДАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ПОЧТОВОГО ОТДЕЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ ПАКЕТА «1С: ПРЕДПРИЯТИЕ»

А.Н. Пылькин, М.А. Клепиков, Д.М. Пантюшина
Рязанский государственный радиотехнический университет имени В.Ф. Уткина,
Российская Федерация, Рязань, m.klepikov62@gmail.com

Аннотация. В работе рассматривается создание программного обеспечения при помощи использованием пакета «1С: Предприятие» на примере почтового отделения.
Ключевые слова: программное обеспечение, «1С: Предприятие».

CREATING SOFTWARE FOR A POST OFFICE USING THE «1C: ENTERPRISE»

A.N. Pylkin, M.A. Klepikov, D.M. Pantyushina
Ryazan State Radio Engineering University named after V.F. Utkin,
Russia, Ryazan, m.klepikov62@gmail.com

Annotation. This paper discusses the creation of software using the "1C: Enterprise" package on the example of a post office.
Keywords: software, "1C: Enterprise".

В настоящее время существует программное обеспечение, позволяющее быстро и эффективно получать логистическую информацию различного вида. Однако для каждого предприятия чаще всего нужна индивидуальная программа, которая содержит в себе конкретные функции. Система программ «1С: Предприятие» предназначена для решения широкого спектра задач автоматизации учета и управления, стоящих перед динамично развивающимися современными предприятиями. Пакет «1С: Предприятие» представляет собой систему прикладных решений, построенных по единым принципам и на единой технологической платформе [1]. Именно из-за предлагаемых функций и удобного интерфейса для реализации программы была выбрана среда «1С: Предприятие».

Целью исследования является автоматизация учета почты с конкретными задачами.

В текущей версии программы пользователь может заполнять 3 типа документов: Накладная, Перевоз Товаров и Зарплата. Анализ предметной области приводит к необходимости создания справочников, регистров сведений и накоплений, перечисления, форм печати и отчетов. В качестве примера интерфейса, используемого в процессе создания документа «Накладная», приведена соответствующая форма на рисунке 1.

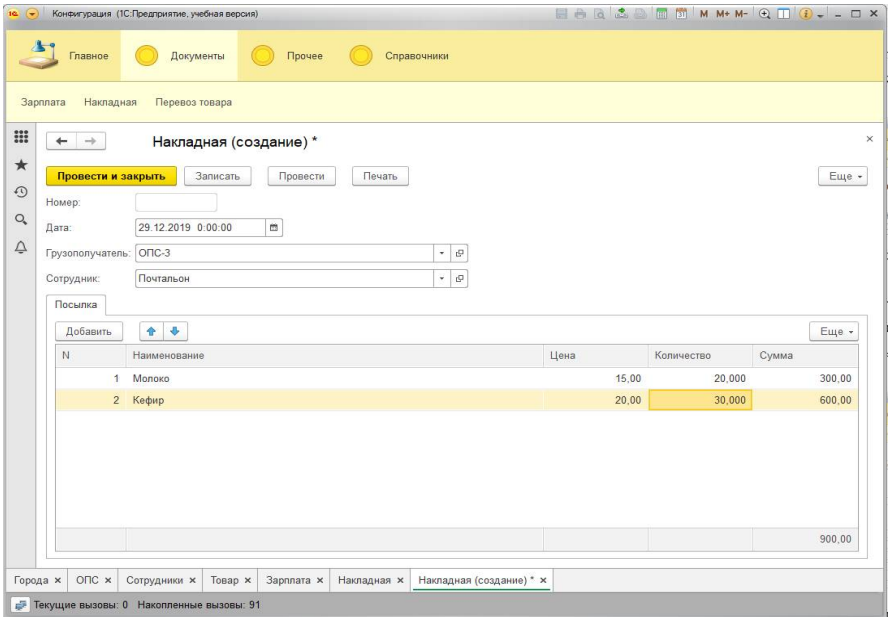


Рис. 1. Создание документа

Если требуется создание отчёта по заданным пользователем временным промежуткам, то печать документа реализована в соответствующем разделе (рис. 2)

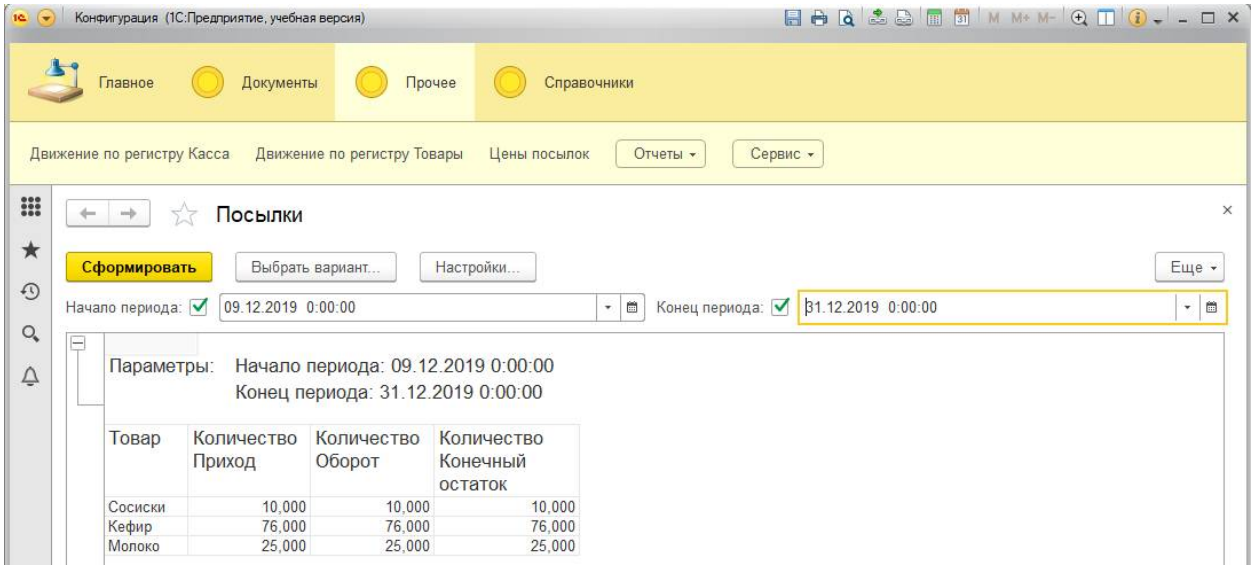


Рис. 2. Создание отчёта

Для правильного функционирования данной программы необходимо было сделать регистр накоплений, который позволяет учитывать прибыль и убытки от произведённых операций. Например, в программе реализован регистр, который из общей суммы средств вычитает расходы на зарплаты, а также учитывает доходы от перевозок, что показано на рисунке 3.

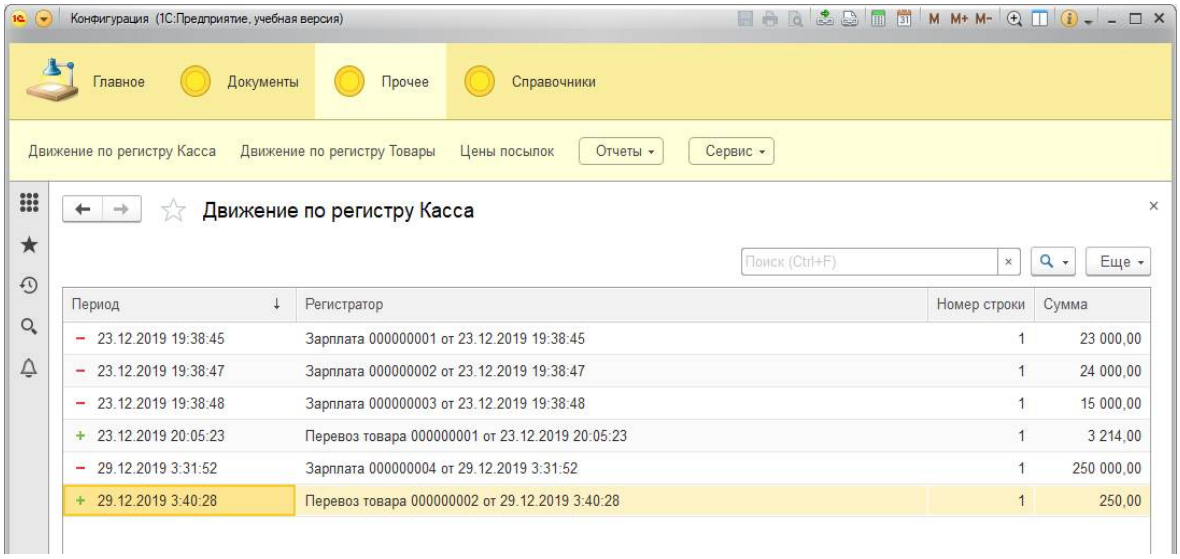


Рис. 3. Регистр накоплений

В данной работе реализована так же функция печати, которая позволяет без дополнительного софта выполнять простейшие функции при работе. Образец формы печати представлен на рисунке 4.

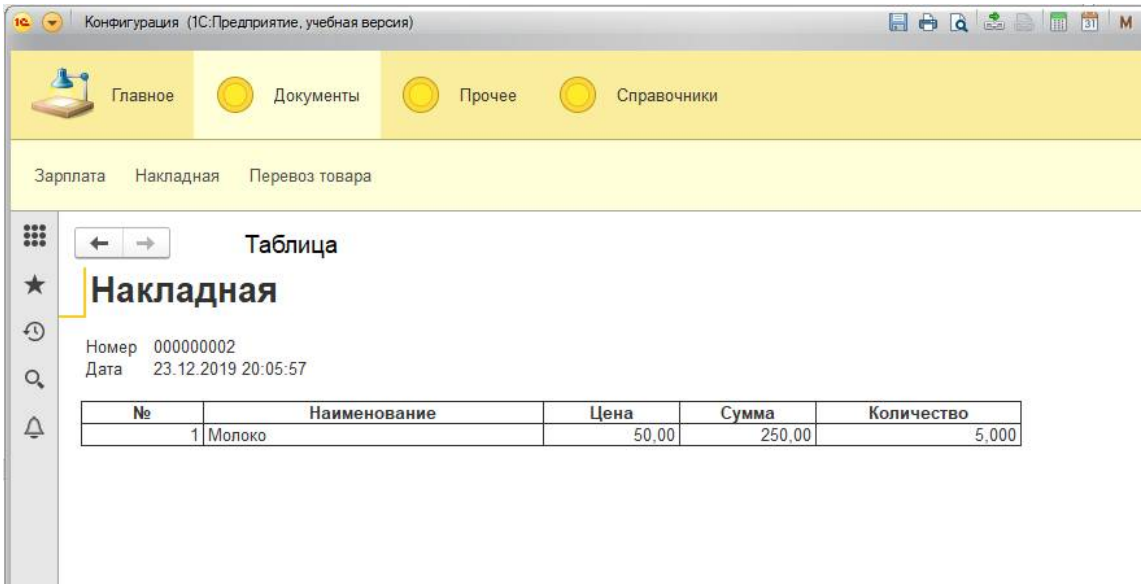


Рис. 4. Форма печати

Рассматриваемое программное обеспечение удобно тем, что позволяет пользователю в простом и удобном интерфейсе создавать необходимые документы. Для реализации более удобной технологии требуется создание расширенного функционала, который будет включать в себя более сложные документы. Предполагаемое программное решение реализовано в рамках деятельности конкретного филиала АО «Почта России» (в том числе при формировании бухгалтерской отчётности различных форм). Планируется создать единый отчёт со всей информацией о перевозке товара по схеме, изображённой на рисунке 5.

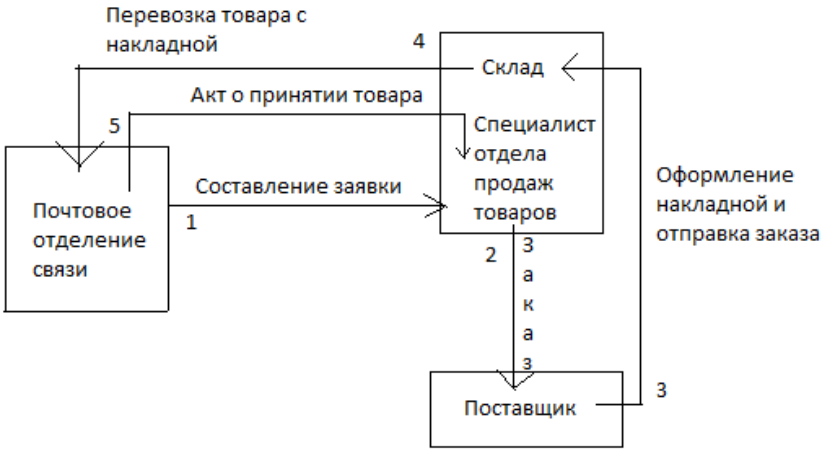


Рис. 5. Схема перевозки

1. 1С: Предприятие [Электронный ресурс] - https://1c.ru/rus/products/1c/predpr/why-predpr.htm - Дата доступа: 14.02.2020.

ОБЗОР МЕТОДА СОПРЯЖЕННЫХ ГРАДИЕНТОВ И ЕГО ОСНОВНЫХ МОДИФИКАЦИЙ ДЛЯ РЕШЕНИЯ НЕЙРОСЕТЕВЫХ ЗАДАЧ

А.В. Шолохов*, П.А. Корнев, А.Н. Пылькин*****
Рязанский государственный радиотехнический университет им. В.Ф. Уткина,
*Россия, Рязань, *renaissan7e@gmail.com, **k-p-al@yandex.ru, ***pylkin.a.n@rsreu.ru*

Аннотация. Рассматриваются различные модификации метода сопряженных градиентов, отличающиеся различным видом реализации параметра сопряженности β_k, предложенных различными авторами. Оценивается сходимость различных алгоритмов с целью выбора оптимальной модификации метода сопряженных градиентов для корректировки весовых коэффициентов в процессе обучения нейросети для задач определения аномальных сетевых пакетов.
Ключевые слова: метода сопряженных градиентов, модификации, параметр сопряженности β_k, нейросетевые задачи.

OVERVIEW OF CONJUGATE GRADIENT METHOD AND ITS MAIN MODICATIONS TO SOLVE NEURAL NETWORK PROBLEMS

A.V. Sholohov*, P.A. Kornyev, A.N. Pylkin*****
Ryazan State Radio Engineering University named after V.F. Utkin,
*Russia, Ryazan, *renaissan7e@gmail.com, **k-p-al@yandex.ru, ***pylkin.a.n@rsreu.ru*

Absract. Various modifications to the conjugate gradient method proposed by various authors are considered. The convergence of different algorithms is evaluated in order to select the optimal modification of the method of conjugate gradients to correct weights in the process of neural network training for tasks of determination of abnormal network packets.
Keywords: modification method of conjugate gradients, conjugation parameter β_k, neural network tasks.

Метод сопряженных градиентов, предложенный в начале 1950-х годов Хестинсом и Штифелем, эффективно применяется для решения задач безусловной минимизации [1]. Данный метод является одной из модификаций метода градиентного спуска с отличием в том, что на каждом шаге в качестве направления спуска используется не антиградиент, а его линейная комбинация с прежним направлением спуска. Таким образом, поиск решения квадра-

тичной задачи происходит за конечное число шагов [2], равное количеству переменных при любом начальном приближении, а его сходимость существенно лучше. Также к достоинствам данного метода можно отнести его простоту и низкие затраты памяти, что делает его особенно эффективным при решении задач большой размерности.

Общая задача безусловной минимизации выглядит следующим образом:

$$f(x) \to min_{x \in R_n}, \tag{1.1}$$

где функция $f: R_n \to R$ непрерывно дифференцируема и ограничена снизу, $g(x)$ – градиент целевой функции.

В общем виде вычислительная схема метода сопряженных градиентов описывается следующим образом:

Нулевой шаг. Возьмем произвольное начальное приближение $x_1 \in R_n$ и вычислим значение градиента $g_1 = g(x_1)$. Если $g_1 = 0$, то x_1 – решение задачи. Иначе задаем направление градиента $d_1 = -g_1$, т.е. по антиградиенту, как в методе градиентного спуска.

k-й шаг. Следующее приближение x_{k+1} вычисляется по формуле:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{1.2}$$

где α_k – приближенное решение задачи одномерной минимизации, $f(x_k + \alpha d_k) \to \min_{\alpha>0}$.

Далее найдем значение градиента $g_{k+1} = g(x_{k+1})$. Если $g_{k+1} = 0$, то x_{k+1} – решение задачи. Иначе вычисляем следующее направление d_{k+1} по формуле:

$$d_{k+1} = -g_{k+1} + \beta_k d_k \tag{1.3}$$

где β_k – параметр сопряженности. Варианты данного параметра представлены в таблице 1.

Таблица 1. Варианты параметра β_k, где $y_k = g_{k+1} - g_k$

Аналитический вид параметра β_k	Изобретатель, год
$\beta_k^{HS} = \frac{\langle g_{k+1},\, y_k \rangle}{\langle d_k,\, y_k \rangle}$	Хестенс, Штифель (1952 г.)
$\beta_k^{FR} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2}$	Флетчер, Ривз (1964г.)
$\beta_k^{PR} = \frac{\langle g_{k+1},\, y_k \rangle}{\|g_k\|^2}$	Полак, Рибьер (1969 г.)
$\beta_k^{CD} = \frac{\|g_{k+1}\|^2}{\langle -d_k,\, g_k \rangle}$	Флетчер (1987 г.)
$\beta_k^{LS} = \frac{\langle g_{k+1},\, y_k \rangle}{\langle -d_k,\, g_k \rangle}$	Лиу, Стори (1991 г.)
$\beta_k^{DY} = \frac{\|g_{k+1}\|^2}{\langle d_k,\, y_k \rangle}$	Дай, Юань (1999 г.)
$\beta_k^{HZ} = \left\langle y_k - 2d_k \frac{\|y_k\|^2}{\langle d_k,\, y_k \rangle}, \frac{g_{k+1}}{\langle d_k,\, y_k \rangle} \right\rangle$	Хагер, Жанг (2005 г.)

Поиск останавливается при достижении требуемой степени сходимости в точке минимума. Иначе определяются новые сопряженные направления и процесс поиска продолжа-

ется до тех пор, пока не будет обеспечена сходимость, или не будет произведен поиск по всем направлениям.

В методе сопряженных градиентов новое и старое направления неортогональны. В случае минимизации положительно определенной квадратичной формы с матрицей Q все направления спуска d_i оказываются Q ортогональными, т.е. удовлетворяют условию $(d_i, Q_{d_i}) = 0$, при любом $i \neq j$. Такие векторы называются сопряженными, отсюда и название метода [3].

Без существенного усложнения фазы обследования окрестности промежуточных точек метод сопряженных градиентов, так же как и метод градиентного спуска может «застревать» в точках локальных минимумов и в оврагах. Первая проблема нивелируется применением метода Флетчера – Ривса. Вторая (в том числе и для минимизации сложных овражных функций с искривленным дном оврага) – применением овражного метода сопряженных градиентов [4], параметр сопряженности которого вычисляется следующим образом:

$$\beta_k = \frac{\langle y_k, g_{k+1} \rangle}{\langle y_k, d_k \rangle} \tag{1.4}$$

Рассмотрим поведение различных вариантов метода сопряженных градиентов на примере функции Стайблински-Танга, которая задается выражением:

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i. \tag{1.5}$$

Глобальный минимум данной функции достигается в точке $m \approx -2.9$, используем ее при $n = 2$ для большей наглядности результатов. Производилось сравнение методов Флетчера-Ривза (FR), Хестенса-Штифеля (HS), Полака-Рибьера (PR), Хагера-Жанга (HZ), результаты представлены на рисунке 1. Остановка метода происходила на k итерации при условии $\|g_k\| < 10^{-6}$.

Таблица 2. Количество итераций

Метод	Число итераций
Флетчера-Ривза (FR)	86
Хестенса-Штифеля (HS)	15
Полака-Рибьера (PR)	15
Хагера-Жанга (HZ)	22

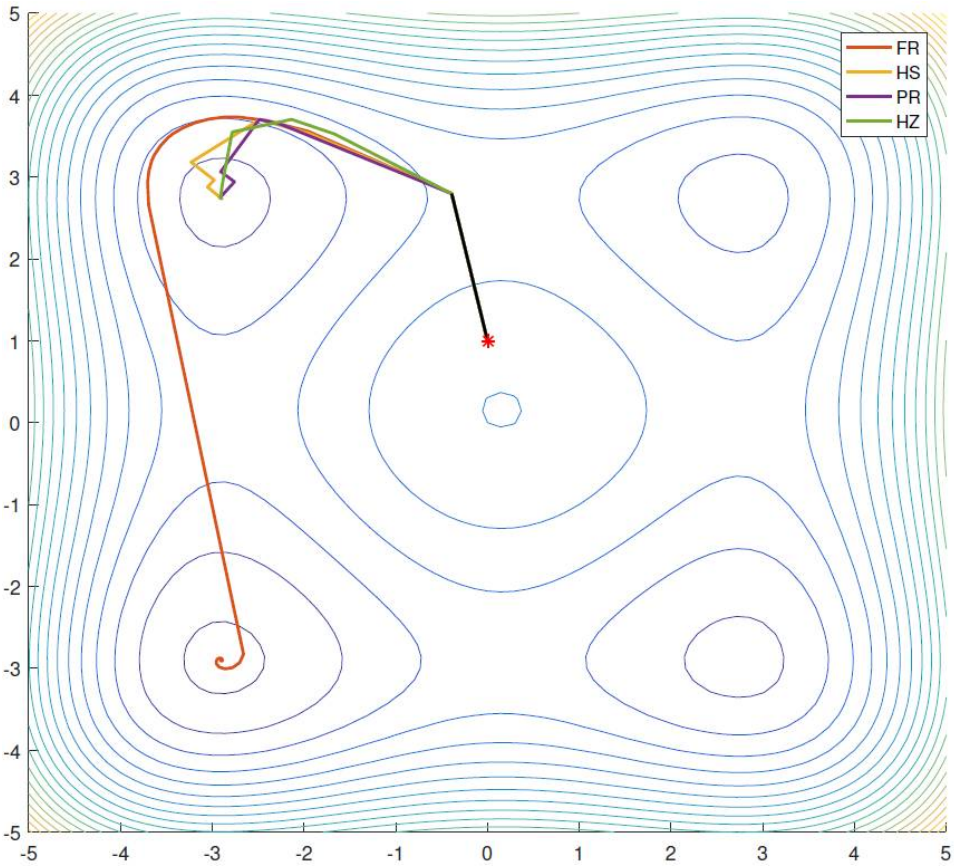


Рис. 1. Сравнение модификаций метода сопряженных градиентов при начальном приближении (0.1, -1)

На данный момент проводится экспериментальное исследование обучения нейронной сети для задач определения аномальных сетевых пакетов [5]. Нейроструктурная модель данной сети гетерогенна, для обучения используется алгоритм многократного распространения ошибки вследствие нестабильности процесса обучения из-за использования различных функций активации (для входного и выходного слоев используются функции активации колоколообразного и сигмоидального типов соответственно, для скрытого слоя – функция типа «И»).

Цель данного исследования – выбор оптимальной модификации метода сопряженных градиентов (параметра β_k) для корректировки весовых коэффициентов в процессе обучения. Это позволит существенно уменьшить среднеквадратическую ошибку обучения и ускорить данный процесс в целом. Результаты экспериментов будут использованы в диссертационной работе.

Библиографический список

1. Hestenes M. R., Stiefel E. Methods of conjudate gradients for solving linear systems. // J. Res. Nat. Bur. Standarts. 1952. Vol. 49. No. 6. P. 409 – 436.

2. Аоки М. Введение в методы оптимизации: Пер. с англ. – М.: Наука, 1977. – 343 с.

3. Струченков В.И. Методы оптимизации в прикладных задачах / Струченков В.И. – М.: Экзамен, 2005. – 256с.

4. Ларичев О.И., Горвиц Г.Г. Методы поиска локального экстремума овражных функций. – М.: Наука, 1990. – 94 с.

5. Корнев П.А., Шолохов А.В. Нейроструктурное моделирование систем искусственного интеллекта для решения сетевых проблем. – ПИТ - 2016: Труды Международной научно-технической конференции / под ред. С.А. Прохорова – Самара: Издательство Самарского научного центра РАН, 2016. – 1052 с. ISBN 978-5-93424-758-5.

СОДЕРЖАНИЕ

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ

Научное издание

В 10 томах

Том 4

Под общей редакцией О.В. Миловзорова.

Подписано в печать 15.06.20. Формат 60х84 1/8.
Бумага офсетная. Печать офсетная.
Гарнитура «Times New Roman».
Усл. печ. л. 24,75.
Тираж 100 экз. Заказ №.

Рязанский государственный радиотехнический университет,
Редакционно-издательский центр РГРТУ,
390005, г. Рязань, ул. Гагарина, д. 59/1.
Отпечатано в типографии Book Jet,
390005, г. Рязань, ул. Пушкина, д. 18