ISSN 2220-3397

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Программная инженерия

ПрИн
6
2018
Том 9

XIX Всероссийская конференция молодых учёных по математическому моделированию и информационным технологиям

г. Кемерово
29 октября – 2 ноября 2018 г.

Научные направления:

- **Математическое моделирование**
- **Численные методы и методы оптимизации**
- **Высокопроизводительные распределенные вычисления**
- **Информационные и геоинформационные системы**
- **Цифровая экономика**
- **Управление, обработка, защита и хранение информации**
- **Автоматизация и теория управления**

Председатель программного комитета
академик Ю.И. Шокин

Прием тезисов до 11 сентября
Регистрация на сайте http://conf.nsc.ru/ym2018

Ученый секретарь конференции
к.т.н. Сергей Александрович Рылов
Телефон: +7-383-334-9173
E-mail: ymconf@ict.sbras.ru

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с сентября 2010 г. DOI 10.17587/issn.2220-3397 ISSN 2220-3397

Журнал издается при поддержке Отделения математических наук РАН, Отделения нанотехнологий и информационных технологий РАН, МГУ имени М.В. Ломоносова, МГТУ имени Н.Э. Баумана

СОДЕРЖАНИЕ

Журнал зарегистрирован в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций. Свидетельство о регистрации ПИ № ФС77-38590 от 24 декабря 2009 г.

Журнал распространяется по подписке, которую можно оформить в любом почтовом отделении (индексы: по каталогу агентства "Роспечать" — **22765**, по Объединенному каталогу "Пресса России" — **39795**) или непосредственно в редакции.
Тел.: (499) 269-53-97. Факс: (499) 269-55-10.
Http://novtex.ru/prin/rus E-mail: prin@novtex.ru
Журнал включен в систему Российского индекса научного цитирования и базу данных RSCI на платформе Web of Science.
Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

THEORETICAL AND APPLIED SCIENTIFIC AND TECHNICAL JOURNAL

SOFTWARE ENGINEERING

PROGRAMMNAYA INGENERIA

Vol. 9
N 6
2018

Published since September 2010 **DOI 10.17587/issn.2220-3397** **ISSN 2220-3397**

CONTENTS

Information about the jornal is available online at:
http://novtex.ru/prin/eng e-mail:prin@novtex.ru

УДК 004.4, 004.94, 004.7 DOI: 10.17587/prin.9.243-252

Д. А. Басавин[1], ассистент кафедры, e-mail: basavind@gmail.com,
С. В. Поршнев[2], д-р техн. наук, проф., e-mail: sergey_porshnev@mail.ru,
Д. А. Петросов[1], канд. техн. наук, доц., зав. кафедрой, e-mail: scorpionss2002@mail.ru,
[1] Белгородский государственный аграрный университет им. В. Я. Горина, п. Майский, Белгородская обл.,
[2] Уральский федеральный университет имени первого Президента России Б. Н. Ельцина, г. Екатеринбург

Стратегии организации вычислительных процессов гибридной жидкостной модели

Обсуждаются подходы к балансировке нагрузки на гетерогенную вычислительную среду, состоящую из центральных и графических процессоров, использованные авторами при разработке параллельной программной реализации гибридной жидкостной модели интернет-трафика (информационных потоков) в современных компьютерных сетях. Опыт разработки и использования параллельной программной реализации гибридной жидкостной модели на основе технологии GPGPU показывает, что время расчета характеристик информационных потоков в высокоскоростных компьютерных сетях оказывается значительно зависящим от распределения вычислительной нагрузки на компоненты гетерогенной архитектуры. Обсуждаются результаты экспериментальных исследований программных реализаций гибридной жидкостной модели, в которых были реализованы различные стратегии обмена данными между компонентами системы.

Ключевые слова: *гетерогенная вычислительная система, интернет-трафик, компьютерные сети, гибридная жидкостная модель интернет-трафика, технология GPGPU, балансировка нагрузки*

Введение

В настоящее время технология *GPGPU*[1] широко применяется при разработке различных параллельных программных средств. Анализ опыта ее использования для разработки программного обеспечения (ПО), в котором реализуются алгоритмы параллельных вычислений, показывает, что одной из основных задач, которые при этом приходится решать разработчикам ПО, является обеспечение сбалансированного распределения вычислительной нагрузки между вычислительными ядрами *GPGPU*. Без ее решения в подавляющем большинстве случаев не удается обеспечить требуемой производительности соответствующего ПО [1].

Необходимость решения отмеченной выше задачи была обнаружена в процессе разработки параллельной реализации гибридной жидкостной модели (ГЖМ) [2], описывающей интернет-трафик в магистральных высокоскоростных компьютерных сетях в терминах изменения во времени скорости передачи информации потоков и длин очередей на входах в соответствующие каналы связи [3]. Подробный анализ организации структур данных, использованных при создании параллельной программной реализации ГЖМ на основе технологии *GPGPU*, проведен в работе [4], где обоснованы выбор типов структур данных, алгоритмов их обработки, методика расчета объемов памяти, необходимой для хранения параметров ГЖМ, и получены оценки максимально возможного числа моделируемых *TCP*-потоков. Эти оценки подтвердили возможность кардинального по сравнению с последовательной программной реализацией ГЖМ увеличения числа одновременно моделируемых информационных потоков и, соответственно, актуальность разработки соответствующей параллельной программной реализации данной модели. Анализ алгоритмов обработки выбранных структур данных, реализуемых на гетерогенном вычислительном устройстве, показал, что они могут быть разделены на два класса:

[1] *GPGPU* (англ. *general-purpose computing for graphics processing units*, неспециализированные вычисления на графических процессорах) — техника использования графического процессора видеокарты, который обычно имеет дело с вычислениями только для компьютерной графики, чтобы выполнять расчеты в приложениях для общих вычислений, которые обычно проводит центральный процессор.

1) алгоритмы, допускающие полное распараллеливание вычислений [5];

2) алгоритмы, допускающие распараллеливание вычислений с ограничениями [6]. При использовании алгоритмов, относящихся ко второму классу, вполне естественно возникает необходимость решения задачи балансировки вычислительной нагрузки в гетерогенной системе, состоящей из центрального (*CPU*) и графического (*GPU*) процессоров.

В настоящей работе обсуждаются подходы к решению задачи балансировки нагрузки на гетерогенную вычислительную систему, реализованные авторами в процессе разработки параллельной программной реализации ГЖМ. Проводится анализ возможных стратегий организации вычислительных процессов в гетерогенной вычислительной среде с точки зрения обеспечения максимальной скорости вычислений и максимального числа моделируемых информационных потоков. В связи с тем что аналогичные задачи приходится решать многим разработчикам, создающим ПО на основе использования технологии *GPGPU*, предложенные авторами решения могут представлять интерес для широкого круга разработчиков данного типа ПО.

Гибридная жидкостная модель интернет-трафика в высокоскоростных компьютерных сетях

Компьютерная сеть в ГЖМ, следуя работе [7], представляется графом $G = (V, E)$, вершины которого V — хосты (узлы, маршрутизаторы), а ребра E — каналы связи между ними (рис. 1).

Здесь ребра графа, соответствующие каналам связи, обозначаются индексом $l \in E$, эти каналы описываются следующими параметрами:

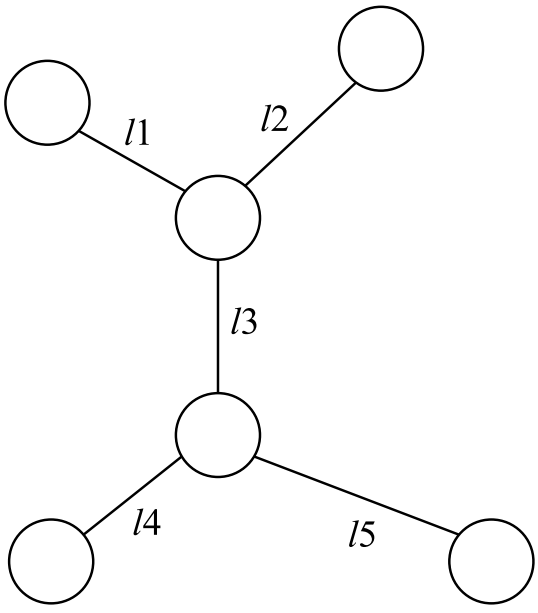


Рис. 1. Структурная модель фрагмента компьютерной сети

- пропускной способностью C_l (бит/с или пакет/с);
- длиной очереди (числом пакетов) на входе в данный канал q_l;
- принятой и реализуемой политикой управления очередью на входе в канал передачи данных, а именно — на маршрутизаторе, определяемой функцией $p_l(t)$ — вероятностью сброса пакетов, зависящей от уровня загрузки (перегрузки) канала в данный момент времени;
- временем передачи пакетов по каналу a_l (время задержки).

В рамках обсуждаемой ГЖМ принимается, что в компьютерной сети (КС) циркулируют потоки данных, которые в случае передачи по одному маршруту и близости их характеристик объединяются в группы или классы. При этом все потоки одной группы рассматриваются как единый поток, интенсивность которого равна сумме интенсивностей отдельных потоков. Для обозначения классов потоков используется индекс $i = 1, \ldots, N$, где N — общее число моделируемых потоков.

Пусть $F_i = \{k_{i,1}, \ldots, k_{i,m_i}\}$ и $O_i = \{j_{i,1}, \ldots, j_{i,m'_i}\}$ — упорядоченный перечень (список) каналов и очередей соответственно, по которым последовательно проходит i-й поток в прямом (передача данных) и в обратном (передача подтверждений) направлениях. Полный маршрут i-го потока данных в прямом и обратном направлениях $E_i = F_i \cup O_i$. Для канала $k \in E_i$ очереди на выходе и на входе k-го канала, в котором распространяется i-й поток, обозначим $s_i(k)$ и $b_i(k)$ соответственно.

Тогда изменения во времени скорости передачи данных i-го потока $W_i(t)$ (размер окна данных) и длины очереди на входе в l-й канал $q_l(t)$ описываются следующей системой однородных линейных уравнений (СОДУ):

$$\frac{dW_i(t)}{dt} = \frac{\theta_{(W_i(t)-M_i)}}{R_i(t)} - \frac{W_i(t)}{2}\lambda_i(t), \tag{1}$$

$$\frac{dq_l(t)}{dt} = -\theta_{q(t)}C_l + \sum_{i \in N_l} A_i^l(t), \tag{2}$$

где $\theta_{f(t)}$ — функция Хевисайда

$$\theta_{f(t)} = \begin{cases} 1, & \text{если} \quad f(t) \geqslant 0, \\ 0, & \text{если} \quad f(t) < 0, \end{cases}$$

$R_i(t)$ — время оборота[2] i-го потока, рассчитываемое по формуле

$$R_i(t) = \sum_{l \in E_i} \left(a_l + q_l / C_l\right), \tag{3}$$

[2] Согласно стандарту протокола *TCP* $R_i(t)$ — сумма времени прохождения потока данных от момента начала их передачи источником до момента их получения приемником и времени от момента отправления приемником подтверждения о получении данного сообщения до его получения передатчиком.

суммирование здесь проводится по всей трассе прохождения i-го потока E_i;

$\lambda_i(t)$ — скорость потери пакетов i-го потока, вычисляемая по формуле

$$\lambda_i(t) = p\left(t - R_i(t)/2\right) A_i\left(t - R_i(t)/2\right); \quad (4)$$

C_l — пропускная способность l-го канала, который обслуживается данным маршрутизатором;

$A_i^l(t)$ — скорость передачи i-го потока по l-му каналу,

$$A_i^l(t) = \frac{W_i^l(t)}{R_i^l(t)}. \quad (5)$$

Из формулы (4) видно, что в жидкостной модели и ГЖМ учитывается, что подтверждение о получении данных, отправленных в момент времени $t - R_i(t)$, поступает на источник данных в момент времени t. Как следствие, в соответствии с протоколом *TCP* в течение данного временного интервала источник отправляет число пакетов, равное текущему размеру окна данных, продолжение передачи новых данных возможно только после подтверждения получения всех отправленных пакетов при текущем значении окна данных.

Из формул (1)—(5) видно, что алгоритм расчета размеров окон передачи данных каждого из потоков (1) может быть полностью распараллелен, так как входящие в выражение (1) значения численных параметров и функций для каждого потока рассчитываются независимо друг от друга, в то время как алгоритм расчета очередей на входе в соответствующие каналы (2) может быть распараллелен, однако с ограничениями ввиду наличия в (2) суммарной скорости передачи данных по каналу $\sum_{i \in N_l} A_i^l(t)$, где $A_i^l(t)$ вычисляется в соответствии с формулой (5). В связи с этим при создании параллельной программной ГЖМ было предложено использовать соответствующие структуры данных, описанные в работе [4]. Одновременное использование полностью распараллеливаемого алгоритма и алгоритма, распараллеливаемого с ограничениями, определяется необходимостью использования соответствующих стратегий организации вычислительных процессов в гетерогенной вычислительной среде.

Анализ возможных стратегий организации вычислительных процессов

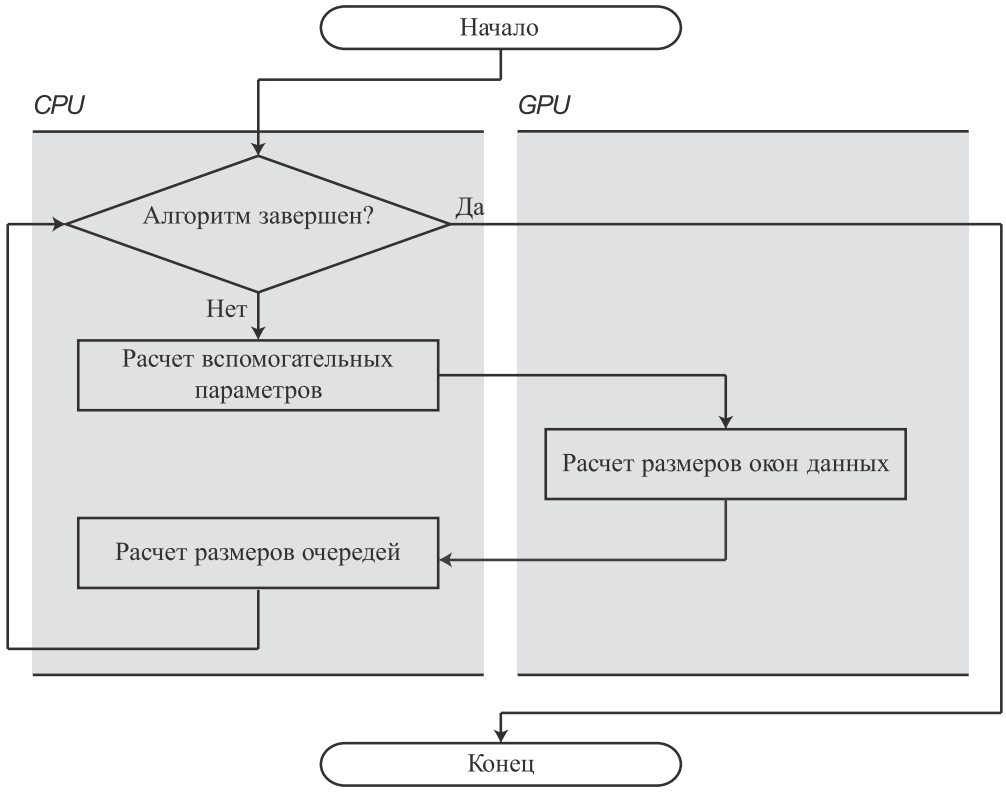
Обоснование возможности использования технологий параллельных вычислений при создании программной реализации ГЖМ приведено в работе [3], где отмечено, что в зависимости от выбора тех или иных "подходящих" для распараллеливания этапов вычислительного алгоритма ГЖМ необходимо выбирать соответствующие стратегии организации вычислительных процессов, которые можно характеризовать накладными расходами, обусловленными необходимостью обмена данными и их синхронизацией между *CPU* и *GPU*, что может негативно сказаться (и сказывается) на общей производительности системы. В связи с этим были проведены соответствующие эксперименты, в которых получены оценки быстродействия возможных стратегий организации вычислительных процессов.

В ходе проведенного исследования были выявлены следующие возможные стратегии организации вычислительных процессов: перенос ресурсоемких полностью распараллеливаемых участков на *GPU* (стратегия 1), перенос всех полностью распараллеливаемых вычислений на *GPU* (стратегия 2), перенос всех вычислений на *GPU* (стратегия 3), непрерывный обмен данными между *CPU* и *GPU* (стратегия 4). Блок-схемы и псевдокод алгоритмов этих стратегий представлены на рис. 2—5.

Стратегия 1 подразумевает перенос на *GPU* только тех этапов ГЖМ, которые хорошо поддаются распараллеливанию и при этом имеют достаточную алгоритмическую сложность, например, такие как независимый расчет параметров на основе входных данных. К таким этапам относятся расчет размера окон данных для каждого потока на каждом шаге моделирования. Здесь можно ожидать повышение производительности за счет наиболее эффективного применения технологии *GPGPU*.

Стратегия 2 представляет собой усовершенствованную стратегию 1 за счет переноса всех полностью распараллеливаемых этапов алгоритма ГЖМ на *GPU*. Данная стратегия позволяет перенести большую часть алгоритмов на *GPU*. Это позволяет ожидать увеличение его использования, что потенциально может сказаться на повышении общей производительности системы.

Стратегия 3 заключается в переносе всех шагов алгоритма, в том числе и распараллеливаемых с ограничениями, на *GPU*, что фактически сокращает число операций обмена данными между процессорами до двух (загрузка и выгрузка), а время, затрачиваемое на эти операции, можно считать константой, независящей от времени моделирования. Данная стратегия позволяет учесть отмеченные выше особенности алгоритма ГЖМ, состоящие в наличии как полностью распараллеливаемых этапов вычислений, так и этапов вычислений, распараллеливаемых с ограничениями. Однако при этом необходимо учитывать, что объем обрабатываемых данных зачастую превышает доступный объем памяти *GPU* (см., например, [4]), что, как очевидно, ограничивает их максимальный размер. Отметим, что потенциально данный проблемный вопрос можно решить за счет использования буферизации и обмена данными между вычислительными устройствами не на каж-



а)

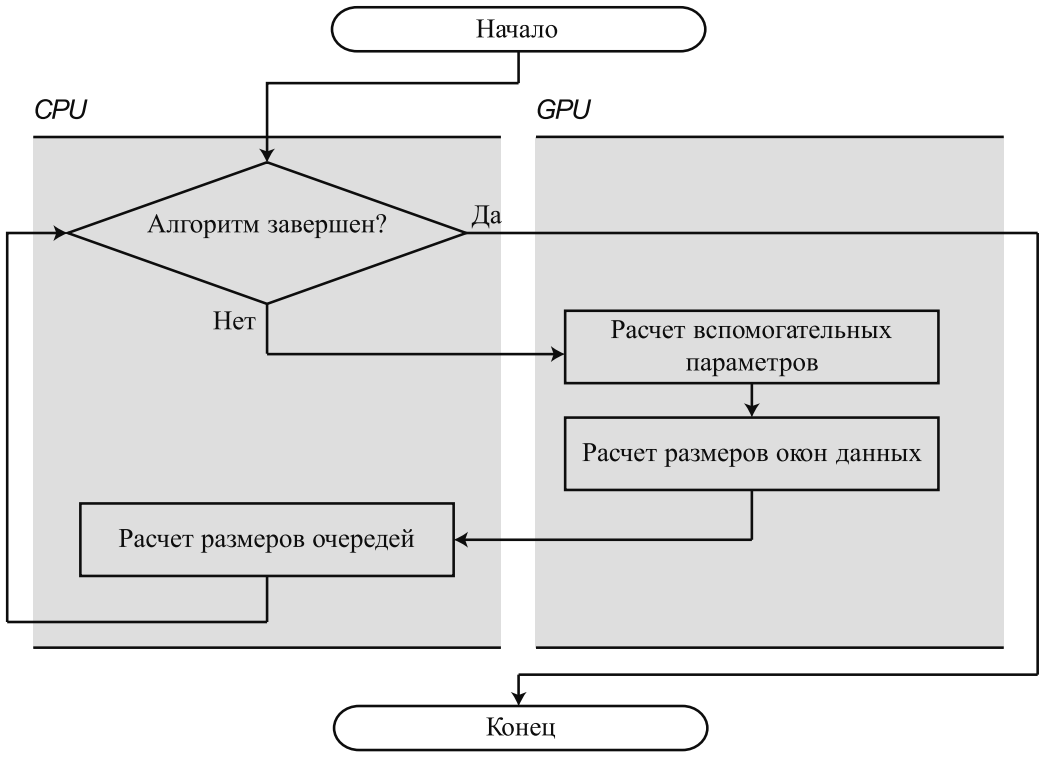
```
Цикл пока время моделирования не закончено:
      Рассчитать вспомогательные параметры ГЖМ
      Загрузить данные в память GPU
      [GPU] Рассчитать размеры окон данных ГЖМ
      Выгрузить данные на CPU
      Рассчитать размеры очередей ГЖМ
Конец цикла
```

б)

Рис. 2. Блок-схема (*а*) и псевдокод (*б*) стратегии 1

дой итерации основного алгоритма, но на отдельных шагах алгоритма, выбранных в зависимости от доступного размера буфера.

Стратегия 4 отличается от стратегий 1—3 тем, что здесь на каждом шаге алгоритма ГЖМ проводятся загрузка и выгрузка необходимых данных из *CPU* в *GPU* и обратно. Потенциально данная стратегия в сравнении со стратегиями 1—3 может обеспечить моделирование значительно более сложных КС. Однако в процессе передачи данных между *CPU* и *GPU*, который, собственно, требует конечного времени, с неизбежностью блокируется процесс вычислений, проводимых как на *CPU*, так и на *GPU*, так как в противном случае могут возникать ошибки, связанные с поступлением неверных данных, используемых на соответствующих шагах алгоритма ГЖМ. При этом по объективным причинам будет снижаться скорость вычислений, проводимых программной реализацией данной стратегии. В этой ситуации, принимая во внимание, что операции обмена данными между *CPU* и *GPU* должны быть реализованы при вычислении значений каждой из функций, использованной в ГЖМ, и каждого узла КС и каждого информационного потока (по оценке не менее 10—12 операций обмена на 1 мкс работы вычислительного ядра) было принято решение отказаться от использования стратегии 4 в дальнейших исследованиях.



а)

```
Цикл пока время моделирования не закончено:
     Загрузить данные в память GPU
     [GPU] Рассчитать вспомогательные параметры ГЖМ
     [GPU] Рассчитать размеры окон данных ГЖМ
     Выгрузить данные на CPU
     Рассчитать размеры очередей ГЖМ
Конец цикла
```

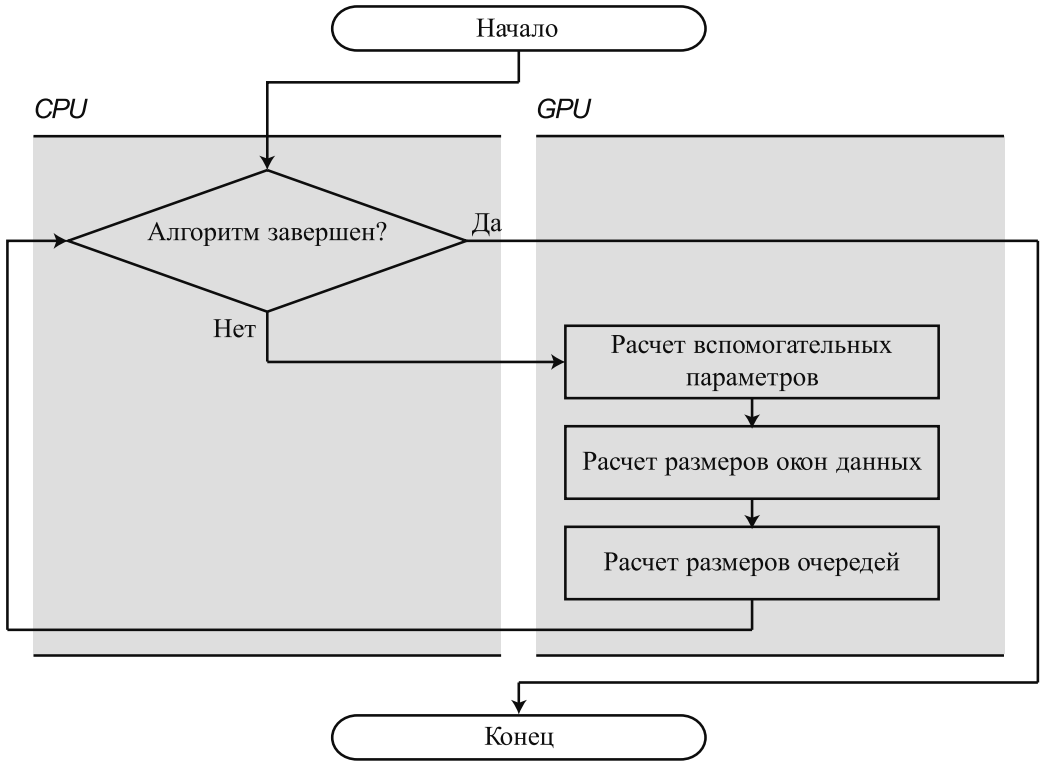
б)

Рис. 3. Блок-схема (*а*) и псевдокод (*б*) стратегии 2

Методика проведения экспериментов

В ходе экспериментальных исследований быстродействия программных реализаций ГЖМ, реализующих соответствующие стратегия обмена данными между *CPU* и *GPU*, было проведено моделирование информационных потоков в КС, концептуальная схема топологии которой представлена на рис. 6.

На рис. 6 видно, что в проведенных экспериментах топология сети представляла собой совокупность групп небольших сетей, каждая из которых являлась подсетью (сегментом), состоящим из двух информационных потоков, одного коммутационного узла (маршрутизатора), одного приемника и трех каналов связи. Общее число сегментов сети для k-го эксперимента составляло $m = 2^{k-1}$, $k = 1, ..., 10$. Здесь каждый *TCP*-источник генерировал нагрузку на протяжении всего времени моделирования, что позволило обеспечить максимальную нагрузку на вычислительные алгоритмы и аппаратные ресурсы. В экспериментах варьировались следующие параметры ГЖМ: число моделируемых *TCP*-источников интернет-трафика, число маршрутизаторов, число приемников. Остальные параметры ГЖМ оставались неизменными. Данный подход позволил проводить идентичные эксперименты с помощью каждой из программных реализаций ГЖМ.



а)

```
Загрузить данные в память GPU
Цикл пока время моделирования не закончено:
      [GPU] Рассчитать вспомогательные параметры ГЖМ
      [GPU] Рассчитать размеры окон данных ГЖМ
      [GPU] Рассчитать размеры очередей ГЖМ
Конец цикла
Выгрузить данные на CPU
```

б)

Рис. 4. Блок-схема (*а*) и псевдокод (*б*) стратегии 3

В k-м эксперименте использовали следующие параметры ГЖМ:

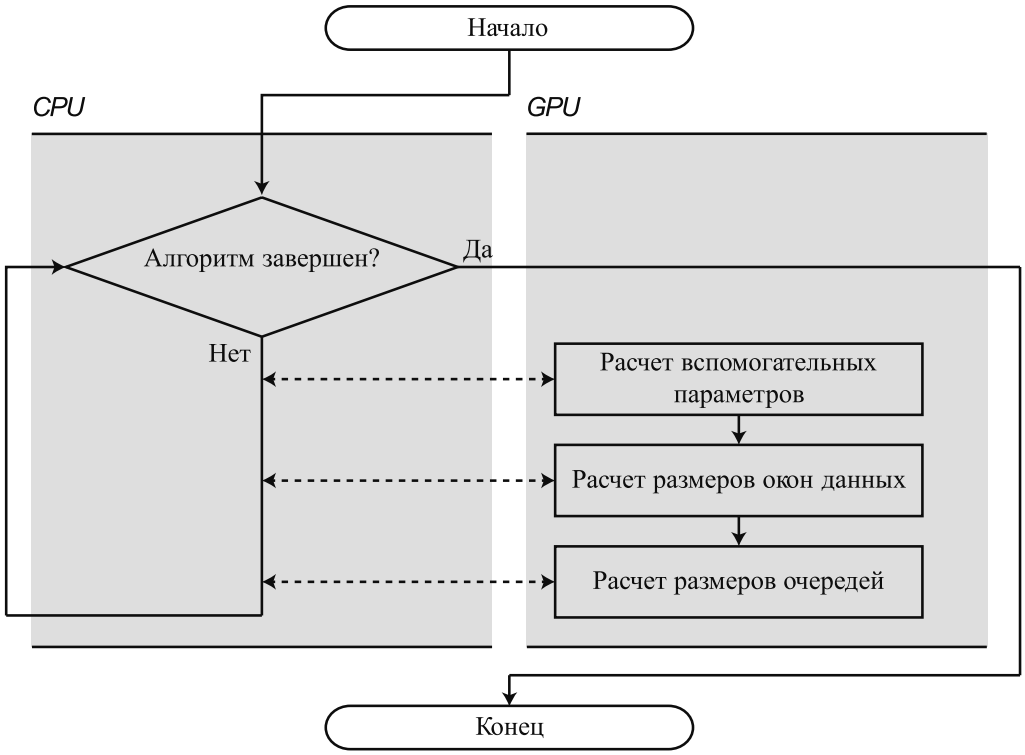
- время моделирования — 1000 мс;
- число пользователей — $n = 2^k$;
- число маршрутизаторов — $m = 2^{k-1}$.

В каждом из экспериментов использовали идентичные параметры ГЖМ. Измерения времени вычислений проводили перед началом моделирования и после его окончания. Для каждой стратегии проводили 20 независимых расчетов и измеряли время их выполнения. Далее для каждой из стратегий вычисляли среднее время, затраченное на вычисления. При этом для исключения влияния внешних факторов на производительность каждой из программных реализаций в экспериментах использовали одинаковое аппаратное обеспечение и операционную систему.

Анализ результатов экспериментов

Зависимости среднего времени расчетов от количества сегментов моделируемой КС для каждой из описанных выше стратегий организации вычислительных процессов представлены на рис. 7.

На рис. 7 видно, что однозначно лучшей стратегии распределения нагрузки между *CPU* и *GPU* нет.



а)

```
Цикл пока время моделирования не закончено:
      Загрузить данные в память GPU
      [GPU] Рассчитать вспомогательные параметры ГЖМ
      Выгрузить данные на CPU
      Загрузить данные в память GPU
      [GPU] Рассчитать размеры окон данных ГЖМ
      Выгрузить данные на CPU
      Загрузить данные в память GPU
      [GPU] Рассчитать размеры очередей ГЖМ
      Выгрузить данные на CPU
Конец цикла
```

б)

Рис. 5. Блок-схема (*а*) и псевдокод (*б*) стратегии 4

Стратегии 1 и 2 имеют более высокую скорость расчетов в случае, когда среднее число моделируемых сегментов относительно невелико (2^3—2^8). При числе моделируемых сегментов, превышающем 2^8, производительность обеих стратегий резко уменьшается. Данный результат объясняется тем, что при небольших объемах обрабатываемых данных *CPU* оказывается эффективнее *GPU*, и наоборот, на больших объемах данных производительность *CPU* оказывается ниже производительности *GPU*, производительность стратегий 1 и 2 начинает заметно ухудшаться. Незначительная разница в производительности программных реализаций стратегий 1 и 2 обусловлена разным числом вычислительных операций, перенесенных с *CPU* на *GPU*.

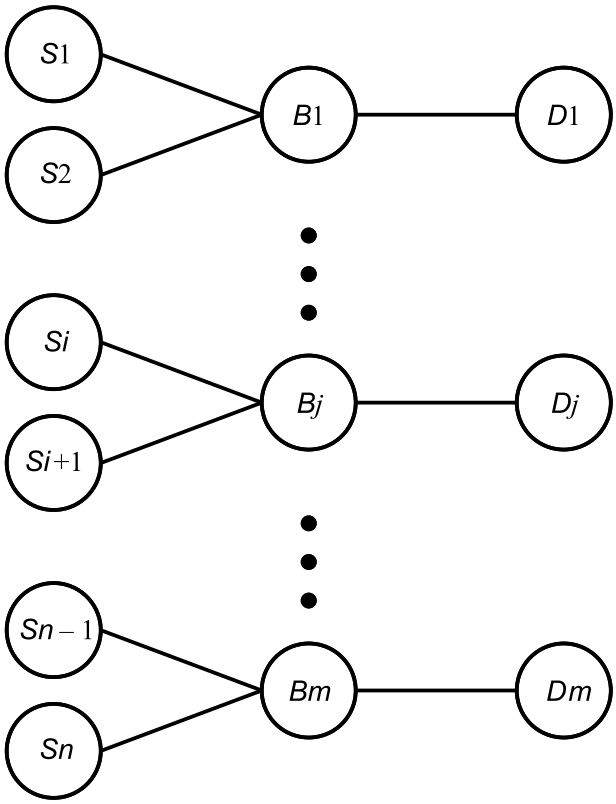


Рис. 6. Концептуальная схема топологии моделируемой сети для эксперимента *k*:

S — TCP-источник интернет-трафика (генератор нагрузки); *B* — коммутационный узел (маршрутизатор); *D* — приемник нагрузки

Скорость работы программной реализации стратегии 3 оказывается выше аналогичного значения программных реализаций стратегий 1 и 2 при небольшом (менее 2^3) и большом (более 2^9) числе сегментов, моделируемой КС, однако оказывается ниже при обработке средних объемов данных (число сегментов сети находится в диапазоне от 2^3 до 2^9). Данный результат свидетельствует о том, что стратегия 3 оказывается наиболее предпочтительной с точки зрения балансировки нагрузки в указанных диапазонах значений числа сегментов сети.

Подобные выводы согласуются с результатами работы [8], свидетельствующими о низкой эффективности расчетов, проводимых с помощью технологии *GPGPU*, в сравнении с аналогичными расчетами на центральных процессорах для небольших объемов данных, и напротив, о заметно более высокой эффективности применения технологии GPRPU для больших объемов данных.

Заключение

В ходе проведения исследования была подтверждена гипотеза о повышении производительности гетерогенной вычислительной среды при разработке программной реализации ГЖМ при полном переносе вычислений на графический процессор. Проведенные эксперименты показали более равномерный рост времени вычислений в зависимости от числа моделируемых сегментов и преимущество переноса всех вычислений на графический процессор, в том случае, когда число моделируемых сегментов КС превышает 2^9.

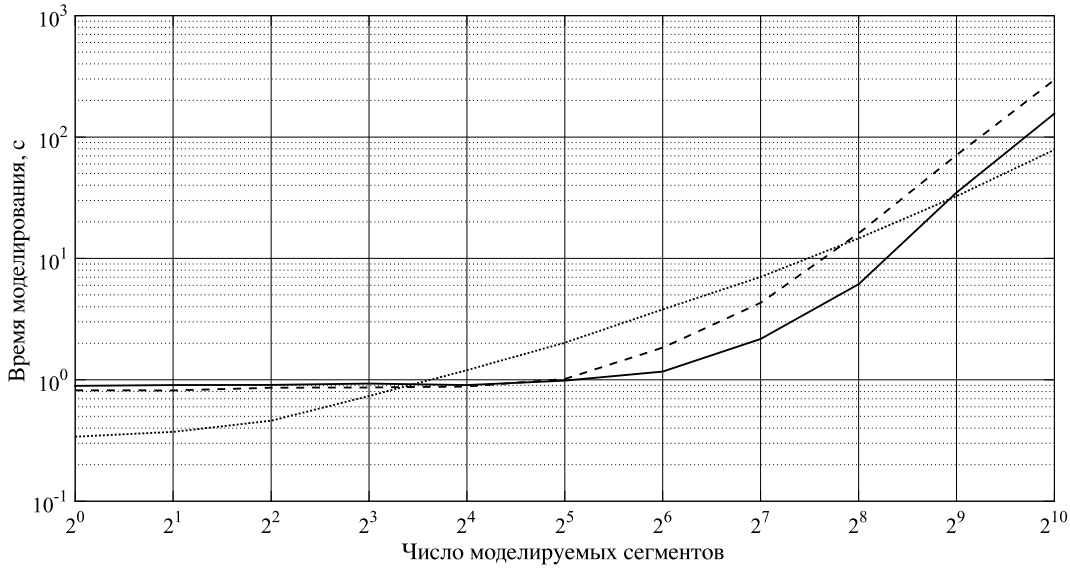


Рис. 7. Зависимость среднего времени расчетов от числа сегментов моделируемой КС:
штриховая линия — стратегия 1; пунктирная линия — стратегия 2; сплошная линия — стратегия 3

Список литературы

1. **Кривов М. А., Притула М. Н., Гризан С. А., Иванов П. С.** Оптимизация приложений для гетерогенных архитектур. Проблемы и варианты решения // Информационные технологии и вычислительные системы. 2012. № 3. С. 72—81.
2. **Misra V., Gong W.-B., Towsley D.** Stochastic Differential Equation Modeling and Analysis of TCP Windowsize Behavior // Proceedings of IFIP WG 7.3 Performance, November, 1999. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.9562&rep=rep1&type=pdf
3. **Басавин Д. А., Поршнев С. В.** Параллельная гибридная жидкостная модель высокоскоростных информационных потоков в магистральных интернет-каналах // Естественные и технические науки. 2013. № 1. С. 317—326.
4. **Басавин Д. А., Поршнев С. В., Петросов Д. А.** Оценка максимального числа моделируемых информационных потоков с помощью параллельной программной реализации гибридной жидкостной модели интернет-трафика на основе технологии *GPGPU* // Программная инженерия. 2017. Т. 8, № 5. С. 195—206.
5. **Басавин Д. А., Поршнев С. В.** Структуры данных и способы их параллельной обработки в рамках задачи моделирования интернет-трафика // Проблемы управления, обработки и передачи информации. Сб. тр. IV Междунар. науч. конф., Саратов, 20—22 сентября 2015 г. В 2-х т. Саратов: Райт-Экспо, 2015. Т. 1. С. 197—202.
6. **Basavin D., Porshnev S.** Problems of Developing Parallel S of Complex Architecture Based on GPGPU Techno Hybrid Fluid-Based Model of Internet Traffic in Computer Networklogy and their Solutions // Proceedings — 2015 International Conference On Computational Intelligence And Communication Networks, 2015. Jabalpur, 12—14 December 2015. IEEE, 2015. P. 156—160.
7. **Liu Y., Presti F. L., Misra V., Towsley D., Gu Y.** Fluid Models and Solutions for Large-Scale IP Networks // ACM SIGMETRICS Performance Evaluation Review. 2003. Vol. 31, Issue 1. P. 91—101.
8. **Basavin D., Porshnev S.** Serial and Parallel Implementations of Hybrid Fluid Model of Information Flows in Networks with Complex Topology // ITM Web Conf. ITM Web of Conferences. 2017. Vol. 10. No. 04001.

Strategies of Parallel Hybrid Fluid Model Computing Algorithms

D. A. Basavin[1], basavind@gmail.com, **S. V. Porshnev**[2], sergey_porshnev@mail.ru,
D. A. Petrosov[1], scorpionss2002@mail.ru,
[1] Belgorod State Agricultural University named after V. Gorin, Mayskiy, Belgorod region, 308503, Russian Federation,
[2] Ural Federal University named after the first President of Russia B. N. Yeltsin, Ekaterinburg, 620002, Russian Federation

Corresponding author:

Basavin Dmitry A., Assistant, Belgorod State Agricultural University named after V. Gorin, Mayskiy, Belgorod region, 308503, Russian Federation
E-mail: basavind@gmail.com

Received on March 14, 2018
Accepted on March 26, 2018

The article discusses strategies of load balancing in a heterogeneous computing environment that includes the central processing unit (CPU) and the graphics processing unit (GPU). The authors used these strategies during development of a hybrid fluid model (HFM) of information flows in modern computer networks with complex topologies. The HFM represents system of ordinary differential equations (ODE) from math point of view and a balance equation of informational flows that comes in and out from a corresponding node of a modeling computer network from physical point of view. Due to the lack of analytical solutions for systems of ODE, which are included in the HFM, it is necessary to develop appropriate software tools that allow making numerical solutions of ODE system in acceptable time.

It was noticed that a computation time of modeling of information flows characteristics in high-speed computer networks significantly depends on distribution of a computational load between components of the heterogeneous architecture. This assumption was formed during development and using of the parallel HFM software implementation based on the general-purpose computing for graphics processing units (GPGPU) technology. Due to this fact there is a need to choose a data exchange strategy between the random access memory (RAM) and the GPU. It should consider HFM computing algorithms details and provide balanced distribution of the computational load between

components of the heterogeneous computing system. This paper describes the results of experimental researches of data exchange strategies.

Keywords: *heterogeneous computing system, Internet traffic, computer networks, parallel hybrid fluid model, modeling, GPGPU, load balancing*

For citation:

Basavin D. A., Porshnev S. V., Petrosov D. A. Strategies of Parallel Hybrid Fluid Model Computing Algorithms, *Programmnaya Ingeneria,* 2018, vol. 9, no. 6, pp. 243—252.

DOI: 10.17587/prin.9.243-252

References

1. **Krivov M. A., Pritula M. N., Grizan S. A., Ivanov P. S.** Optimizacija prilozhenij dlja geterogennyh arhitektur. Problemy i varianty reshenija (Optimize applications for heterogeneous architectures. Problems and solutions), *Informacionnye tehnologii i vychislitel'nye sistemy*, 2012, no. 3, pp. 72—81 (in Russian).

2. **Misra V., Gong W.-B., Towsley D.** Stochastic Differential Equation Modeling and Analysis of TCP Windowsize Behavior, *Proceedings of IFIP WG 7.3 Performance*, November, 1999, available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.9562&rep=rep1&type=pdf

3. **Basavin D. A., Porshnev S. V.** Parallel'naja gibridnaja zhidkostnaja model' vysokoskorostnyh informacionnyh potokov v magistral'nyh internet-kanalah (The parallel hybrid fluid model of high-speed information streams in the backbone Internet channels), *Estestvennye i tehnicheskie nauki*, 2013, no. 1, pp. 317—326 (in Russian).

4. **Basavin D. A., Porshnev S. V., Petrosov D. A.** Ocenka maksimal'nogo chisla modeliruemyh informacionnyh potokov s pomoshh'ju parallel'noj programmnoj realizacii gibridnoj zhidkostnoj modeli internet-trafika na osnove tehnologii GPGPU (Estimation of Maximum Number of Simulated Information Flows for Parallel Software Implementation of a Hybrid Fluid Model of Internet Traffic using GPGPU), *Programmnaya Ingeneria*, 2017, vol. 8, no. 5, pp. 195—206 (in Russian).

5. **Basavin D. A., Porshnev S. V.** Struktury dannyh i sposoby ih parallel'noj obrabotki v ramkah zadachi modelirovanija internet-trafika (Data structures and methods of their parallel processing within the Internet traffic modeling problem), *Problemy upravlenija, obrabotki i peredachi informacii. Sbornik trudov IV Mezhdunarodnoj nauchnoj konferencii*, Saratov, 20—22 September 2015, Saratov, Rajt-Jekspo, 2015, vol. 1, pp. 197—202 (in Russian).

6. **Basavin D., Porshnev S.** Problems of Developing Parallel S of Complex Architecture Based on GPGPU Techno Hybrid Fluid-Based Model of Internet Traffic in Computer Networklogy and their Solutions, *Proceedings — 2015 International Conference on Computational Intelligence and Communication Networks,* IEEE, 2015, pp. 156—160.

7. **Liu Y., Presti F. L., Misra V., Towsley D., Gu Y.** Fluid Models and Solutions for Large-Scale IP Networks, *ACM SIGMETRICS Performance Evaluation Review*, 2003, vol. 31, no. 1, pp. 91—101.

8. **Basavin D., Porshnev S.** Serial and Parallel Implementations of Hybrid Fluid Model of Information Flows in Networks with Complex Topology, *ITM Web of Conferences,* 2017, vol. 10, number 04001.

ИНФОРМАЦИЯ

УДК 004.415.25 DOI: 10.17587/prin.9.253-261

Д. А. Юхимец, д-р техн. наук, ст. науч. сотр., e-mail: undim@iacp.dvo.ru,
Институт автоматики и процессов управления ДВО РАН, г. Владивосток,
Э. Э. Юдинков, магистрант, e-mail: anstertum@gmail.com, Дальневосточный федеральный университет, г. Владивосток

Разработка прикладного программного интерфейса для управления роботом-манипулятором Mitsubishi RV-2FB*

Предложен и обоснован подход, реализующий на основе веб-технологий с использованием протоколов HTTP и TCP/IP распределенную систему управления манипулятором Mitsubishi RV-2FB. Используемый для этого интерфейс позволяет управлять таким манипулятором без задания жесткой программы перед началом выполнения технологических операций. Такой подход предоставляет возможность включения интерфейса в интеллектуальные очувствленные промышленные комплексы.

Ключевые слова: *манипулятор, программный интерфейс, HTTP, распределенное управление, TCP/IP, управление движением*

Введение

Разработка средств и методов программного управления сложными техническими системами является одной из наиболее актуальных задач, решаемых при создании принципиально новых робототехнических комплексов. При этом, как правило, все современные промышленные робототехнические системы предназначены для решения стандартных задач автоматизации производственных процессов в промышленности и лишены внешних программных интерфейсов. Разработка таких интерфейсов требует использования специализированного программного обеспечения конкретного производителя оборудования. Это обстоятельство не позволяет сопрягать их с внешним программным обеспечением и решать нестандартные задачи автоматизации. Также решения, как правило, предполагают использование интеллектуальных алгоритмов планирования действий на основе обработки информации от внешних сенсорных систем. Отмеченная особенность создает большие сложности для разработчиков при создании интеллектуальных производственных участков.

Задача, направленная на устранение отмеченных выше сложностей, активно решается для мобильных роботов различного назначения, бортовое оборудование которых может содержать устройства различных производителей. При этом программное обеспечение, под управлением которого функционируют эти роботы, обеспечивает им уникальные функциональные возможности и в большинстве случаев создается заново. Для решения задачи построения систем управления робототехническими комплексами с широким набором оборудования в настоящее время активно используются представленные далее виды платформ.

ROS (*Robot Operating System*) — это свободное программное обеспечение, реализующее системный уровень управления роботом [1]. При этом основная задача, которую решают с использованием этого программного обеспечения, заключается в повторном использовании кода для реализации функций низкоуровневого управления оборудованием, для передачи сообщений между процессами и реализации часто используемых функций (управление, навигация и т. д.). Использование ROS для реализации систем управления различными робототехническими комплексами описано в работах [2, 3]. Преимуществом использования ROS является наличие большого числа библиотек, обеспечивающих интеграцию с различными бортовыми устройствами. Однако ROS не имеет средств взаимодействия с промышленным оборудованием, что создает сложности использования этого программного обеспечения для построения распределенных производственных систем.

LabVIEW (*Laboratory Virtual Instrument Engineering Workbench*) представляет собой платформу (совокупность программных средств) для системного проектирования и среду разработки с визуальным языком программирования G, которые разработаны компанией National Instruments. Обычно LabVIEW используется для сбора данных, управления приборами и для автоматизации промышленных процессов под различными операционными системами (ОС), включая Microsoft Windows, Unix, Linux и MacOS. Инструментарий LabVIEW построен на базе FRC

* Работа поддержана РФФИ (грант 16-07-00718).

(*FIRST Robotics Competition*), целью которого является создание и программирование роботов для выполнения задач автономного управления. Примеры успешных реализаций различных систем управления и обработки информации на основе LabVIEW описаны в работах [4, 5]. Следует однако отметить, что эта платформа имеет закрытый исходный код, что не позволяет использовать ее с оборудованием, для которого не реализована поддержка.

URBI (*Universal Real-time Behavior Interface*) — открытая программная платформа, которую используют для создания сложных систем управления, работающих в реальном времени [6]. Инструментарий URBI основывается на распределенной компонентной архитектуре, в которой алгоритмы, реализованные на языке C++, встраиваются в специальную интерфейсную оболочку UObject. При этом взаимодействие компонентов UObject описывается на специальном скриптовом языке urbiscript, что обеспечивает гибкую настройку всей системы. Примеры использования URBI для построения систем управления роботов различного назначения представлены в работе [7].

OROCOS (*Open Robot Control Software project*) — открытое программное обеспечение, которое представляет собой набор библиотек для создания систем управления роботами различного типа. Его основное функциональное назначение — реализация алгоритмов решения задач кинематики и динамики робототехнических систем, алгоритмов обработки информации, обеспечение работы в режиме реального времени. Примеры использования OROCOS описаны в работе [8].

Как видно из представленного выше краткого анализа, существующие программные платформы для построения систем управления роботами обеспечивают реализацию промежуточного программного обеспечения (*middleware*), отвечающего за взаимодействие между различными компонентами системы. При этом указанные платформы ориентированы на построение систем управления мобильной робототехникой или объектами, имеющими в своем составе устройства с открытым программным интерфейсом, что редко выполняется для промышленного оборудования. По этой причине появляется необходимость в решении задачи разработки подхода к созданию программного интерфейса для устройств промышленной автоматизации, позволяющего включать эти устройства в системы управления, реализованные с помощью внешних программ.

1. Постановка задачи

В настоящей работе рассматривается подход к созданию программного интерфейса к промышленному манипулятору Mitsubishi RV-2FB. Этот робот управляется специализированным контроллером CR-750D, который обеспечивает: запуск программ движения робота; управление его позиционированием; обработку информации от внешних цифровых датчиков и пульта оператора. Программы для робота Mitsubishi RV-2FB реализуются на специализированном языке MELFA BASIC V и представляют собой последовательность команд, задающих движение рабочего органа робота по последовательности заранее заданных точек. Стандартные средства создания программ для робота не предусматривают изменение координат этих точек в процессе движения робота. Такой подход позволяет использовать рассматриваемый интерфейс только для выполнения действий в заранее известном окружении.

С учетом изложенного выше, в работе, результаты которой приведены в статье, ставится и решается следующая задача. Необходимо разработать программный интерфейс для робота-манипулятора Mitsubishi RV-2FB, обеспечивающий управление этим роботом из внешней программы. При этом разрабатываемый интерфейс должен соответствовать следующим требованиям.

1. Программный интерфейс должен обеспечивать удаленный доступ к функциям робота через локальную сеть и сеть Интернет.

2. Программный интерфейс должен реализовывать все функции и режимы движения робота, которые обеспечивает контроллер, а также обеспечить доступ к информации о текущем состоянии манипулятора.

3. Время реакции робота на поданную через программный интерфейс команду не должно превышать 15 мс. Это требование обеспечивает работу робота в реальном времени.

2. Описание разработанного программного интерфейса для управления манипулятором Mitsubishi RV-2FB

В основе разработанного программного интерфейса к промышленному манипулятору Mitsubishi RV-2FB лежат набирающая популярность программная платформа для разработки веб-приложений Vue.js, а также библиотека сетевого взаимодействия Flask, которая используется в качестве связующего звена между всеми составляющими частями программного интерфейса.

Использование этой платформы позволяет создавать интерактивные и динамические пользовательские интерфейсы, которые удовлетворяют концепции управления в режиме реального времени. При этом использование библиотеки Flask позволяет реализовать низкоуровневую коммуникацию между пользовательским интерфейсом и роботом манипулятором, а также обеспечить устойчивость соединения.

Конечным звеном реализованной системы выступает программа на языке MELFA BASIC V, выполняющаяся на промышленном контроллере CR-750D. Эта программа отвечает за прием команд от программного интерфейса, их сериализацию и запуск средствами контроллера. Обобщенная структурная схема программного взаимодействия изображена на рис. 1.

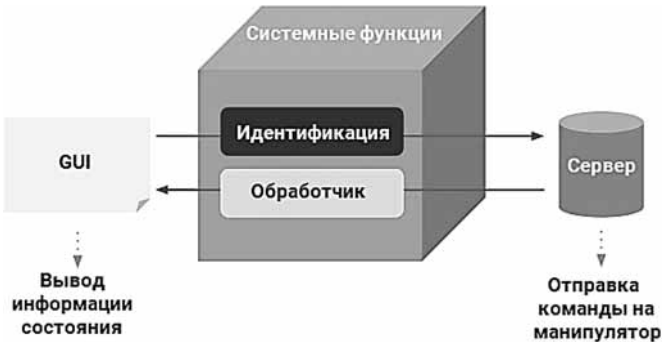


Рис. 1. Обобщенная структурная схема программного взаимодействия

Как видно на рис. 1, программный интерфейс состоит из следующих трех основных частей: пользовательского представления (GUI), реализованного с помощью компонентов, основанных на платформе Vue.js; системных функций, реализованных с помощью библиотеки Flask и отвечающих за передачу данных между компонентами системы; обработчика (сервер), отвечающего за взаимодействие программной части системы и манипулятора.

Пользовательское представление является связующим звеном между пользователем программного интерфейса и сервером на Flask, оно реализует решение следующих задач:

- преобразование данных пользовательского ввода;
- маршрутизация между элементами пользовательского интерфейса;
- отображение актуальной информации состояния манипулятора.

Системные функции и пользовательское представление общаются двунаправленно через связующую функцию, выполняющую роль моста. Этот факт означает, что системные функции могут осуществлять операции запроса к внутренним функциям Vue, что упрощает реализацию GUI. При этом пользовательское представление через HTTP-сервис Axios получает последовательность данных в формате JSON (*JavaScript Object Notation*), несущих информацию о пространственной ориентации рабочего органа манипулятора, и осуществляет операцию рендеринга данных в DOM (*The Document Object Model*) в пределах запущенной сессии управления.

2.1. Системные функции

Набор системных функций также построен на программной платформе Vue.js в качестве отдельного модуля. Их функциональным назначением является: осуществление коммуникации между сервером на Flask и роботом-манипулятором; передача данных в графический буфер для последующей отрисовки в GUI. Центральным элементом набора системных функций является компонент localstorage, работающий на основе шаблона управления состоянием и реализующий Flux-архитектуру [9]. Предложенная

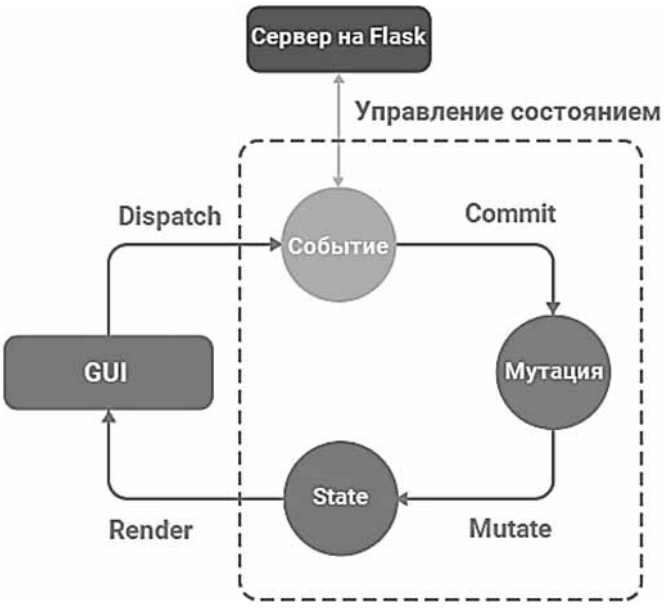


Рис. 2. Шаблон управления состоянием

архитектура позволяет реализовать событийно-ориентированную систему, которая обеспечивает управление состоянием приложения согласно заданному шаблону. Она позволяет использовать любую библиотеку вместе с классом EventEmitter из NodeJS.

Структура шаблона управления состоянием приложения, реализованная с помощью архитектуры диспетчеризации состояний redux, представлена на рис. 2.

Шаблон представляет собой единое дерево состояний приложения (state), которое изменяется с помощью вспомогательных методов управления данными, описываемых в JavaScript-объекте (хранилище). Изменение состояния происходит при получении хранилищем сообщения (action), содержащего данные и описывающего нужные действия.

В качестве центрального узла всего приложения, который получает от пользовательского интерфейса сообщения и рассылает их зарегистрированным хранилищам, выступает диспетчер (dispatcher). Пример инициализации диспетчера представлен на листинге 1.

Описанный пример вызывает метод dispatch, который рассылает сообщения (action) всем зарегистрированным в нем хранилищам, после обработки сообщений в этих хранилищах состояние приложения будет обновлено.

Инициализация шаблона управлением состояния (листинг 2) осуществляется вызовом метода createStore, который по сути является JavaScript-объектом.

Главным свойством этого объекта является свойство state (листинг 3), в котором хранится вся актуальная информация о состоянии робота, а именно: значения координат фланца, углы поворота приводов, а также описательный каркас команд управления.

```
let dispatcher = require('flux').Dispatcher;



let app = new dispatcher();

app.handleAction = function(action) {

   this.dispatch({

      source: 'SOME ACTION',

      action: action

   });

}

module.exports = app;
```

Листинг. 1. Инициализация диспетчера

```
const store = new mUx.createStore({...})
```

Листинг 2. Инициализация шаблона управлением состояния

```
const store = new mUx.createStore({
   state : {
      coordinates: [],
      angles: [],
      commands: {}
   }
})
```

Листинг 3. Свойство state объекта mUx

Согласно API Reference, Flux-архитектура предполагает использование специальных вспомогательных функций, описанных в сообщениях actions и обеспечивающих передачу данных диспетчеру. Сам диспетчер при этом в силу особенностей реализации позволяет выполнять указанные функции асинхронно. В actions описываются JavaScript-функции, задачей которых является отправка запросов к серверной программе Flask. При этом функция, описанная в рамках свойства actions, в качестве входного параметра принимает не только данные (например, текущие координаты робота), но и специальный JavaScript-объект, содержащий state и два зарезервированных метода dispatch и commit (листинг 4).

Метод dispatch используется для того, чтобы запускать функции, описанные в рамках свойства actions (листинг 5). Запускаются они при этом асинхронно, т. е. загрузка данных никак не будет влиять на работу приложения, даже если в процессе загрузки возникла ошибка. Это свойство особенно важно для работы GUI, так как робот постоянно передает различную информацию о своем состоянии, и использование классических методов для считывания этих данных привело бы к возникновению микрофризов в работе интерфейса вследствие остановки главного потока приложения для вызова функции запроса.

Метод commit служит для запуска специально зарезервированных функций, называемых мутациями (mutations). С их помощью можно осуществлять изменение значений свойства state или модернизировать текущее состояние приложения. В качестве примера наиболее часто используемой мутации можно привести мутацию set (листинг 6), которая обновляет значения координат робота в графическом пользовательском интерфейсе.

Однако концепция мутаций плохо применима для данных, которые необходимо изменять лишь в пределах метода, не затрагивая оригинала. Этот факт проявляется, например в ситуациях, когда нужно сформировать новую целевую точку для движения схвата манипулятора. В этом случае, если изменить свойство coordinates объекта state, то указанные изменения сразу отобразятся в пользовательском интерфейсе. Однако это может ввести в заблуждение пользователя, так как координаты манипулятора

```
const store = new mUx.createStore({
   actions : {
      getCoordinates( { state, dispatch, commit }, addr) {
         let data;
         axios.get(addr)
         .then(response => {
           data = response.data
      })
         .catch(e => {
           this.errors.push(e)
         })
       return data;
      }
  })
```

Листинг 4. Свойство actions объекта mUx

изменятся только после выполнения соответствующей команды. Для устранения подобных казусов предлагается использовать концепцию системных функций-заполнителей (геттеров), которые могут использовать данные, содержащиеся в хранилище, и применять к ним какие-либо фильтры (листинг 7).

Функция, описанная в листинге 7, реализует процесс согласования значений текущих координат состояния робота со свойством type объекта commands, в котором содержатся символьные наименования соответствующих значений параметра data (x, y, z и т. д.).

2.2. Взаимодействие с манипулятором

Взаимодействие между элементами GUI и роботом-манипулятором осуществляется с помощью локального HTTP-сервера, созданного с применением стандартных средств библиотеки Flask (листинг 8).

Все поступающие на сервер команды проверяются на корректность и, в случае их правильности, с помощью протокола TCP/IP передаются в контроллер робота-манипулятора. В противном случае команда игнорируется, а робот переводится в режим ожидания новой команды. Пример реализации сервера для обработки запроса представлен на листинге 9.

Таким образом, предложенный подход к построению архитектуры программного интерфейса позволяет реализовать системы распределенного управления, использующие алгоритмы формирования траекторий движения робота в процессе его работы на основе данных, поступающих от внешних сенсорных устройств, а также реализовывать сложные графические человеко-машинные интерфейсы.

3. Апробация полученного решения

Для изучения эффективности разработанного программного интерфейса были проведены исследования качества работы сервера при обработке поступающих запросов и исследования качества работы манипулятора.

Для исследования качества работы сервера использовалась утилита Fiddler, которая позволяет симулировать запросы к маршрутам. В начале формировался одиночный запрос GET на получение системной информации от робота, после которого формировался запрос POST, содержащий в себе координаты новой точки. Результат выполнения запросов представлен в табл. 1.

Как видно из данных табл. 1, единичные запросы не требуют поддерживать соединения открытыми после завершения сессии (keep-alive). При этом для оценки качества межпрограммной коммуникации требуется определить среднее время выполнения одного запроса, которое может быть получено на основе измерения времени выполнения нескольких контрольных запросов. Результаты выполнения трех запросов представлены в табл. 2.

Как следует из данных табл. 2, среднее время выполнения запросов GET и POST составило 11,23 мс, что соответствует заявленным требованиям и позволяет управлять роботом в режиме движения из точки в точку

```
const store = new mUx.createStore({
   dispatch('getCoordinates', addr)
})
```

Листинг 5. Пример вызова метода dispatch

```
const store = new mUx.createStore({
   mutations : {
      set(state, { type, items }) {
         state[type] = items
      }
   },
   commit('set', { type: 'coordinates', items: 'coordinates' })
})
```

Листинг 6. Пример вызова метода commit

```
const store = new mUx.createStore({
    getters : {
      result(state) {
         return state.coordinates.map(object => {
             return object.id + 1
         }
      }
})
```

Листинг 7. Пример реализации функции-заполнителя

```
from flask import Flask, request, jsonify, render_template, abort, flash
app = Flask(__name__)

…

app=sample_app.make_app(config_files=\'development.ini',))
app.run(host='127.0.0.1', port=8080)
```

Листинг 8. Инициализация HTTP-сервера

```
from flask import Flask, request, jsonify, render_template, abort, flash
app = Flask(__name__)

global manipulator, manipulatorConnection
class tcpConnect():

    def __init__(self, port):
        self.port = port

    def createConnection(self):
        print >>sys.stderr, 'Waiting for a robot connection'
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.bind(('', self.port))
        s.listen(1)
 connection, client_address = s.accept()
        print >>sys.stderr, 'Robot connected:', client_address
        return connection

   def sendData(self, connection, data):
        connection.sendall(data)

    def recv(self, connection, bufferSize):
        result = connection.recv(bufferSize)
        return result

manipulator = tcpConnect(10003)
manipulatorConnection = manipulator.createConnection()
try:
    data = manipulator.recv(manipulatorConnection, buffSize)
    runApplication(manipulator, manipulatorConnection, data)
@app.route('/data, methods=['GET', 'POST'])
def dispatcher():
    if request.method == 'POST':
        if not request.form['id']:
            error = 'Unknown command id'
        elif not request.form['type']:
            error = 'Invalid type of command'
        elif not request.form['description']:
            error = 'You have to specify command description'
        else:
            content = jsonify({"id":request.form['id'],
                               "type":request.form['type'],
                              "description":request.form['description']})
            manipulator.sendData(manipulatorConnection, content)
            flash('Command successfully sent')
    return render_template(data.html', error=error)
```

Листинг 9. Обработка запросов на сервере

(point-to-point). Однако в режиме формирования траектории движения запросы к роботу-манипулятору должны отправляться с периодом, соответствующим периоду поступления данных от внешних датчиков. При этом такой период должен быть как можно меньше для того, чтобы обеспечить высокую точность движения робота по формируемой траектории, особенно при движении с большими скоростями.

Уменьшить время запросов можно с помощью переиспользования ранее открытых соединений. Для этого, кроме значащих запросов, необходимо отправлять пустые запросы со специальными заголовками Connection: keep-alive, которые будут поддерживать соединения в открытом состоянии. При этом среднее время обработки таких сообщений составляет 2,33 мс, что гораздо меньше накладных расходов на установление соединения.

Из представленных в табл. 3 данных видно, что при использовании предложенного подхода время одного запроса GET уменьшается в 1,51 раза, т. е. до 7,83 мс, а время запроса POST — в 1,43 раза, т. е до 8,73 мс. Таким образом, среднее время обработки запро-

Таблица 1

Одиночные запросы GET и POST

Тип запроса	Число соединений	Keep-alive	Коэффициент повторного использования соединений
GET	1	0	—
POST	1	0	—
Примечание: прочерк означает, что при одиночных запросах измерить коэффициент нельзя.			

Таблица 2

Контрольные запросы

Номер запроса	Время выполнения, мс
1	11,09
2	11,77
3	10,83

Таблица 3

100 запросов GET

Тип запроса	Число соединений	Keep-alive	Коэффициент повторного использования соединений
GET	100	43	1,51
POST	100	30	1,43

са, рассчитанное по 100 запросам, составляет 8,28 мс, что в большинстве случаев достаточно для качественного управления манипуляционными операциями.

Для проверки возможности создания GUI был реализован интерфейс оператора, представленный на рис. 3 (см. третью сторону обложки). Этот интерфейс частично повторяет функциональные возможности штатного пульта оператора. В нем реализовано два независимых режима управления, а именно — режим управления движением схвата манипулятора в декартовой системе координат (*XYZ*), связанной с основанием робота, и режим управления углами поворота сервоприводов, установленных в его звеньях (JOINT). В интерфейсе отображается также актуальное состояние манипулятора (значения углов поворота сервоприводов или координат и углов ориентации схвата). Помимо управления движением манипулятора реализована возможность управления открытием/закрытием схвата и задания скорости его перемещения. Тестирование разработанного интерфейса оператора показало, что его использование возможно на любом компьютере, включенном в одну локальную сеть с контроллером робота, и не приводит к задержкам в управлении.

Качество управления манипулятором оценивалось в процессе отработки им двух видов траекторий — прямолинейной и гармонической. При этом дополнительно исследовалась надежность процесса управления при передаче большого числа управляющих сообщений.

При исследовании прямолинейного движения задавалась траектория перемещения схвата между двумя точками с координатами (45,00, 93,00, 327,00, 180,00, 0,00, −135,00) и (130,00, 267,00, 327,00, 180,00, 0,00, −135,00) в плоскости *XY*. Здесь первые три координаты задают положение схвата в системе координат, связанной с основанием манипулятора, а оставшиеся координаты являются углами Эйлера, задающими ориентацию схвата манипулятора. Так как робот двигался без использования линейной интерполяции с инкрементом по координате *X* в 0,5 мм, то в случае потери пакета, содержащего целевую точку, на графике траектории движения схвата наблюдалось вместо прямолинейного отрезка некое подобие дуги.

Траектория движения схвата манипулятора, построенная на основе значений его координат, полученных в моменты его движения, представлена на рис. 4.

Как видно на рис. 4, на результирующей траектории движения отсутствуют непрямолинейные участки. Этот факт означает, что все сообщения дошли успешно, что также подтверждается использованием утилиты Fiddler.

Исследовалось качество управления манипулятором при его движении по криволинейной пространственной траектории. В этом случае движение манипулятора в плоскости *YZ* задавалось с помощью изменения координаты *Z* по синусоидальному закону, а координата *Y* при этом менялась с инкрементом 0,5 мм.

Полученная траектория движения схвата робота представлена на рис. 5.

Как видно на рис. 5, видимые разрывы и явные искажения траектории отсутствуют. Этот факт свидетельствует о том, что, как и в предыдущем случае,

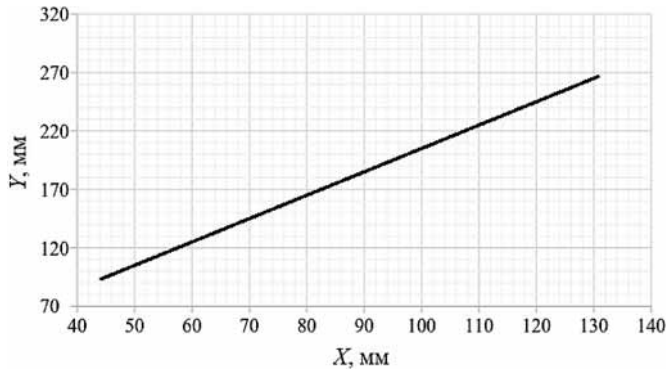


Рис. 4. Траектория движения схвата манипулятора при его движении в плоскости *XY*

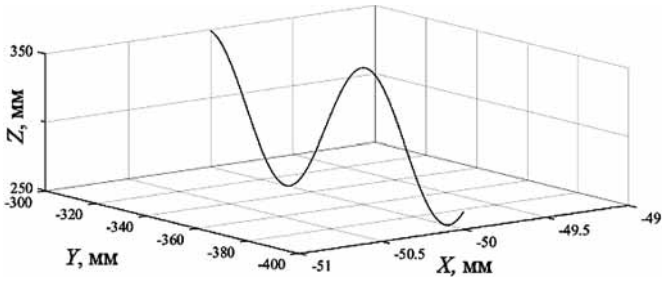


Рис. 5. Траектория движения схвата в плоскости *YZ* по синусоидальному закону

все управляющие сообщения дошли успешно, что также подтверждается утилитой Fiddler.

Таким образом, результаты исследований позволяют утверждать, что реализованный программный интерфейс для робота-манипулятора Mitsubishi RV-2FB показал свою эффективность и надежность при передаче управляющих сообщений. При этом он обеспечивает простое управление движением манипулятора без использования пульта оператора или без необходимости повторного написания программ для контроллера CR-750D.

Заключение

Предложен метод реализации программного интерфейса для промышленного манипулятора на основе использования веб-технологий в целях создания распределенных систем удаленного управления промышленными роботами. Актуальность решаемой задачи определяется необходимостью интеграции стандартного промышленного оборудования с внешними интеллектуальными системами обработки информации и управления для автоматизации промышленных операций в условиях неопределенной рабочей среды. Описана архитектура построения программного интерфейса на основе использования современной программной платформы для разработки веб-приложений Vue.js и библиотеки сетевого взаимодействия Flask. Использование этой платформы позволяет создавать интерактивные и динамические пользовательские интерфейсы, которые удовлетворяют концепции управления в режиме реального времени.

Следует отметить, что предложенный подход не является единственно возможным. Поставленную задачу можно решить с использованием любого высокоуровневого языка программирования с поддержкой сетевого взаимодействия по протоколу TCP/IP. Однако подобная реализация сложнее и будет иметь ряд ограничений пользовательского взаимодействия. Использование же веб-технологий устраняет эти сложности и позволяет сконцентрироваться на логике систем управления, а не на их пользовательских интерфейсах.

Список литературы

1. **Robot** Operating System, Wikipedia, URL: https://en.wikipedia.org/wiki/Robot_Operating_System

2. **Quigley M., Berger E., Ng A.** STAIR: Hardware and Software Architecture, Computer Science Department, Stanford University, Stanford, CA. USA. 2007. URL: http://robotics.cs.brown.edu/aaai07/materials/stanford_paper.pdf

3. **Quigley M., Gerkey B., Conley K., Faust J., Foote T., Leibs J., Berger E., Wheeler R., Ng A.** ROS: an open-source Robot Operating System, Computer Science Department, Stanford University, Stanford, CA. USA. 2009. URL: http://ai.stanford.edu/~ang/papers/icraoss09-ROS.pdf

4. **Sibai F. N., Trigui H., Zanini P. C., AI-Odail A. R.** Evaluation of Indoor Mobile Robot Localization Techniques // Journal of Emerging Trends in Computing and Information Sciences. 2012. Vol. 4. Special Issue ICCSII. URL: http://www.cisjournal.org/journalofcomputing/archive/Vol4SI/Vol4SI_1.pdf

5. **Natarajan N., Aparna S., Sam Jeba Kumar J.** Robot Aided Remote Medical Assistance System using LabVIEW // International Journal of Computer Applications. 2012. Vol. 38, N. 2. P. 6—10.

6. **Universal** Real-time Behavior Interface, Wikipedia, URL: https://en.wikipedia.org/wiki/URBI

7. **Baillie J.-C.** URBI: Towards a Universal Robotic Low-Level Programming Language // Intelligent Robots and Systems, IEEE/RSJ International Conference on. Canada, 2005. P. 820—825.

8. **Lages W. F., Ioris D., Santini D. C.** An Architecture for Controlling the Barrett WAM Robot Using ROS and OROCOS // Conference ISR ROBOTIK 2014. Berlin, VDE VERLAG GMBH. 2014. P. 649—656.

9. **Flux** application architecture for building user interfaces. URL: https://facebook.github.io/flux/docs/overview.html#content

Development of Applied Programming Interface for Control of Manipulator Mitsubishi RV-2FB

D. A. Yukhimets, undim@iacp.dvo.ru, Institute of automation and control processes FEB RAS, Vladivostok, 690041, Russian Federation, **E. E. Yudinkov**, anstertum@gmail.com, Far Eastern Federal University, Vladivostok, 690000, Russian Federation

Corresponding author:

Yukhimets Dmitriy A., Senior Researcher, Institute of automation and control processes FEB RAS, Vladivostok, 690041, Russian Federation
E-mail: undim@iacp.dvo.ru

Received on February 27, 2018
Accepted on March 19, 2018

The implementation method of application programming interface for industrial manipulator on the base of web technologies for creating distributed control systems is proposed in this paper. Topicality of this task is defined by necessity integration of a standard industrial equipment with external intelligent systems of control and data processing for automatization of complex technological operations in uncertain environment.

The application programming interface for robot manipulator Mitsubishi RV-2FB is based on the popular software platform for development of web applications Vue.js and network communication library Flask as connected link between all modules of application interface.

Using this platform allows one to create interactive and dynamic user interfaces which satisfied conception of real-time control. Herewith using Flask library allows one to implement low-level communication between user interface and robot manipulator and provides stability of the connection. Final link of implemented system is the program on MELFA BASIC V language executing by industrial controller CR-750D. This program is responsible for receiving commands from user interface, their serialization and execution by controller resources.

Developed user interface consists of three main parts: user's view implemented by components based on Vue.js platform; systems functions, implemented by Flask library and responsible for data transferring between system components; dispatcher (server), responsible for interacting of software part and manipulator.

Results of research confirm that implemented application programming interface for robot manipulator Mitsubishi RV2-FB has efficacy and reliability for transferring control messages. Also it provides simple control of manipulator movement without using of operators panel or necessity of writing programs for controllers CR-750D.

Keywords: *manipulator, program interface, HTTP, distributed control, TCP/IP, motion control*

Acknowledgements: *This work was supported by the Russian Foundation for Basic Research (project No. 16-07-00718)*

For citation:

Yukhimets D. A., Yudinkov E. E. Development of Applied Programming Interface for Control of Manipulator Mitsubishi RV-2FB, *Programmnaya Ingeneria,* 2018, vol. 9, no. 6, pp. 253—261.

DOI: 10.17587/prin.9.253-261

References

1. **Robot** Operating System, Wikipedia, available at: https://en.wikipedia.org/wiki/Robot_Operating_System

2. **Quigley M., Berger E., Ng A.** *STAIR: Hardware and Software Architecture*, Computer Science Department, Stanford University, Stanford, CA. USA, 2007, available at: http://robotics.cs.brown.edu/aaai07/materials/stanford_paper.pdf

3. **Quigley M., Gerkey B., Conley K., Faust J., Foote T., Leibs J., Berger E., Wheeler R., Ng A.** *ROS: an open-source Robot Operating System,* Computer Science Department, Stanford University, Stanford, CA. USA, 2009, available at: http://ai.stanford.edu/~ang/papers/icraoss09-ROS.pdf

4. **Sibai F. N., Trigui H., Zanini P. C., AI-Odail A. R.** Evaluation of Indoor Mobile Robot Localization Techniques, *Journal of Emerging Trends in Computing and Information Sciences*, 2012, vol. 4, Special Issue ICCSII, available at: http://www.cisjournal.org/journalofcomputing/archive/Vol4SI/Vol4SI_1.pdf

5. **Natarajan N., Aparna S., Sam Jeba Kumar J.** Robot Aided Remote Medical Assistance System using LabVIEW, *International Journal of Computer Applications*, 2012, vol. 38, no. 2, pp. 6—10.

6. **Universal** Real-time Behavior Interface, Wikipedia, available at: https://en.wikipedia.org/wiki/URBI

7. **Baillie J.-C.** URBI: Towards a Universal Robotic Low-Level Programming Language, *Intelligent Robots and Systems*, *IEEE/RSJ International Conference on.*, Canada, 2005, pp. 820—825.

8. **Lages W. F., Ioris D., Santini D. C.** An Architecture for Controlling the Barrett WAM Robot Using ROS and OROCOS, *Conference ISR ROBOTIK 2014,* Berlin, VDE VERLAG GMBH 2014, pp. 649—656.

9. **Flux** application architecture for building user interfaces, available at: https://facebook.github.io/flux/docs/overview.html#content

УДК 004.912 DOI: 10.17587/prin.9.262-271

К. В. Лунев, аспирант, e-mail: kirilllunev@gmail.com, Механико-математический факультет МГУ имени М. В. Ломоносова

Графовые методы определения семантической близости пары ключевых слов и их применения к задаче кластеризации ключевых слов

Представлены результаты исследований на направлении поиска моделей, алгоритмов и программных средств для определения семантической близости между двумя ключевыми словами. Методы, использованные в работе, основаны на теоретико-графовых алгоритмах. Документ представляется в виде множества ключевых слов, ассоциированных с этим документом. Определена мера контекстной близости пары ключевых слов. По заданной коллекции документов строится граф ключевых слов. Вершины этого графа соответствуют ключевым словам, а ребра отражают факт контекстной близости пары слов. Далее представлен метод кластеризации построенного графа. Ключевые слова, входящие в один кластер, обладают свойством семантической близости, что является важным результатом настоящей работы. Программная реализация разработанных моделей протестирована на коллекциях ключевых слов к научным публикациям, а также на коллекции тегов к постам в социальной сети ВКонтакте.

Ключевые слова: *семантическая близость, обработка естественного языка, алгоритмы на графах, теория графов, кластеризация*

Введение

Многие современные информационно-коммуникационные структуры, такие как социальные сети, блоговые и поисковые системы используют ключевые слова для описания содержащихся в них сущностей (объектов). Такой подход значительно упрощает для пользователя поиск необходимых ему объектов системы, так как позволяет сделать это с помощью запроса к системе на естественном языке. К числу таких объектов относятся, например, текстовые документы, изображения, видеозаписи и любой другой объект, которому был приписан набор ключевых слов. Многие исследователи активно занимались и продолжают заниматься анализом ключевых слов в целях кластеризации, визуализации, классификации, индексации и поиска целевых объектов.

Исследования, результаты которых представлены в настоящей работе, затрагивают важную и востребованную практикой задачу кластеризации объектов по ключевым словам, ассоциированным с этими объектами. Ее решение помогает находить в больших информационных коллекциях кластеры похожих объектов, удалять дубликаты документов, определять экспертные сообщества. Далее под мерой (степенью) смысловой близости и похожести (далее — "близость", "схожесть") будем подразумевать показатель семантического сходства пары рассматриваемых ключевых слов или набора слов естественного языка.

В целях анализа эффективности уже существующих и поиска новых подходов к решению рассматриваемой задачи автором проведены библиографические поисковые исследования, результаты которых представлены в настоящей работе. Существует большое число общих методов определения схожести пары слов естественного языка. Их можно разделить на методы, не использующие, и методы, использующие дополнительные источники информации. К числу первых относятся исторически наиболее ранние и наивные подходы, которые вычисляют близость, не используя никакой дополнительной информации о словах, кроме их непосредственного написания. Такие методы основаны на подсчете редакторского расстояния [1, 2], подсчете фонетической близости между словами [3, 4], в этих методах рассматривают слово как множество символов (или как множество символьных n-грамм — последовательностей из n подряд идущих символов) и далее определяют близость между словами как близость между соответствующими множествами [5]. Таким образом, данное направление позволяет строить метрики близости, основанные исключительно на написании конкретных слов. Данный класс методов позволяет достаточно эффективно решать задачу исправления опечаток, но имеет множество недостатков. Основной из них заключается в том, что при

использовании этого класса методов не учитывается семантика слов. Это обстоятельство существенно сужает круг прикладных задач, для решения которых эти методы можно было бы применить.

Несмотря на отмеченный выше недостаток, исследования на этом направлении ведутся до сих пор. Данный класс методов может использоваться в более сложных и совершенных моделях в качестве дополнительных источников для определения близости. Примером более сложной модели в этом направлении является модель редакторского расстояния с настроенными стоимостями для операций вставки, удаления и замены символов [6]. Отметим, что для поиска оптимальных стоимостей при таком подходе необходим дополнительный набор данных с примерами похожих по написанию пар слов.

Существенного улучшения качества определения близости между парой ключевых слов можно добиться, используя дополнительные знания о словах. Это могут быть тексты, в которых слова употреблены, коллекции ключевых слов, информация об объектах, к которым эти слова приписаны, вручную составленные тезаурусы и словари. Каждое из этих направлений имеет свои преимущества и недостатки, анализ которых приведен далее.

Одно из направлений определения близости пар ключевых слов с использованием вспомогательных данных — изучение частот использования рассматриваемых слов в различных коллекциях текстовых документов. Базовым методом определения близости пары ключевых слов в рамках таких исследований является сбор информации о совместной встречаемости слов внутри одного набора. Данные методы детально описаны в работах [7, 8]. Более совершенные на этом направлении алгоритмы основаны на вычислении взаимной информации (*Pointwise mutual information*, PMI), введенной авторами работы [7]. Использование таких алгоритмов позволяет получить решение об уровне близости пар слов не только по их совместной встречаемости, но и путем учета частоты встречаемости каждого из слов в коллекции. Согласно этой метрике высокое значение семантической близости имеют пары слов, которые часто встречаются вместе и редко поодиночке. Работы на этом направлении уже позволяют получать некоторую информацию о семантической близости пары слов. Для этого вводится понятие контекста — некоторого множества слов языка, куда попала пара слов, для которых считается близость. В случае работы с полнотекстовой информацией контекстом могут служить сами текстовые документы (в этом случае проверяется, попали ли оба рассматриваемых слова в один документ), предложения или текстовые n-граммы фиксированной длины.

Недостатком описанных выше статистических методов является необходимость сбора коллекции данных большого размера, поскольку значения PMI и подобных статистических метрик сильно неустойчивы. Частично эта сложность устраняется с помощью перехода от частот слов к вероятностям их появления с последующим применением техник сглаживания для языковых моделей. Этот и другие методы улучшения качества метрики PMI описаны в работах [9—14]. В этих публикациях отмечается, что сам контекст, в котором употребляются слова, используется в самом примитивном виде, а именно в данном контексте проверяется факт наличия обоих рассматриваемых слов. Остальные слова контекста никак не учитываются, что является существенным недостатком описанных выше подходов.

Существует класс методов, решающих задачу семантической близости пар слов с помощью готовых тезаурусов, таких как WordNet [15] и др. [16—22]. Недостаток тезаурусных подходов заключается в их неполноте, а также в том, что некоторые из них не являются публично доступными. Кроме того, тезаурусы обычно охватывают общий домен, и существует мало словарей для специфических областей.

С развитием вычислительной техники большой популярностью начинают пользоваться методы, основанные на обучении нейронных сетей. Одними из самых известных методов определения семантической близости слов являются модели word2vec и GloVe [23, 24]. Суть таких методов в построении векторных представлений для всех слов словаря по контекстам, в которых эти слова употребляются. Данные методы являются очень эффективными, но требуют огромных наборов данных для обучения моделей. Это обстоятельство делает их неприменимыми к задачам, в которых полные тексты документов недоступны. К числу таких задач принадлежит задача определения семантической близости пары ключевых слов научных публикаций. Эти методы зачастую также определяют контекстную близость, по определению которой два слова близки, если они встречаются в похожих контекстах. Во многих практических задачах подобного эффекта использования метрики близости хочется избежать, поскольку, например, слова "математика" и "физика" могут встречаться в одних и тех же контекстах, но как пара ключевых слова для научных публикаций эти слова явно не являются семантически близкими.

Для решения задачи определения близости ключевых слов перспективными являются методы, основанные на теоретико-графовых алгоритмах. В работе [25] в качестве исходных данных рассмотрена коллекция наборов ключевых слов к научным публикациям. Рассмотрен граф, вершинами которого являются ключевые слова, а ребра отражают факт принадлежности пары слов к одному из наборов. Далее по построенному графу для пары вершин вычисляют различные характеристики, включая расстояние, число кратчайших путей, различные графовые меры близости. Недостатком описанной модели является тот факт, что смысловая близость между парой слов стремительно падает при увеличении расстояния в графе. Это происходит по той причине, что набор ключевых слов далеко не всегда состоит из близких по смыслу слов. Последнее обстоятельство приводит к тому, что уровень семантической близости пары

ключевых слов быстро уменьшается с увеличением длины пути между вершинами в графе.

В направлении кластеризации текстовой информации также существует большое число работ (например, [26—29]). Основным ограничением в применении таких методов является использование полнотекстовой информации для построения графов и/или связей между словами. Поэтому для решения задачи кластеризации множества ключевых слов автором настоящей работы был разработан метод кластеризации, основанный на введенных далее по тексту графах близости ключевых слов и на описанном в работе [30] алгоритме кластеризации графов.

В настоящей работе представлены методы определения близости объектов по корпусу наборов характеризующих их ключевых слов, эти методы также опираются на методы из теории графов. Важным направлением в таком подходе к определению близости является получение максимального количества семантической информации о ключевых словах по небольшому набору данных. Значительным улучшением качества определения семантической близости является построение второго графа ключевых слов, основанного на контекстной близости пары слов. Далее в статье дано определение контекстной близости для пары ключевых слов, а также представлены методы построения такого графа. Также представлены алгоритмы семантической кластеризации, основанные на введенной мере близости слов. В разделе "Тестовые испытания" представлены тестовые данные и результаты экспериментов на этих данных программных реализаций разработанных автором алгоритмов.

Построение графа ключевых слов

Вычисление смысловой близости пары ключевых слов основывается на построении графа ключевых слов. Вершины этого графа соответствуют ключевым словам, а взвешенные ребра отражают факт вхождения слов в один набор. Значение веса ребра между вершинами i, j графа G определяется формулой

$$G(i,\ j) = \sum_{\{T|i\in T,\ j\in T\}} \frac{1}{|T|},$$

где суммирование проводится по всем наборам T, которые содержат в себе оба слова — i и j.

Таким образом, если слова часто встречаются в коротких наборах, то вес соответствующего ребра в графе будет высоким. Эта характеристика является важной для определения семантической близости, но не определяющей: слова не обязаны быть похожими друг на друга по смыслу. Напротив, нередко они служат для того, чтобы более точно описать общую тему документа, к которому относятся, а добавление точного синонима не добавляет информации о тематике документа.

В работе [25] по такому графу вычисляли близость ключевых слов. Следует, однако, отметить, что в таком виде граф не дает значительного улучшения результата. Это происходит вследствие того, что ребра между непохожими друг на друга вершинами "зашумляют" значения метрики: высокий уровень метрики начинает больше указывать на уровень употребимости пары слов вместе, чем на семантическую близость между ними. Чтобы избежать подобного эффекта была разработана более эффективная модель вычисления контекстной близости для пары ключевых слов, описанию которой посвящен следующий раздел.

Модель определения контекстной близости для пары ключевых слов

Важной идеей новой модели вычисления смысловой близости по сравнению с моделью, описанной в работе [25], является следующее наблюдение: уровень похожести слов x и y увеличивается, если существует большое число слов k, входящих в одни наборы и с x, и с y. С учетом этого факта была разработана модель определения контекстной близости, описание которой приводится далее. Согласно проведенным тестовым испытаниям контекстная близость является более точной аппроксимацией смысловой близости, чем близость, введенная в работе [25]. Общие слова k в рамках новой модели выступают в роли общего контекста для слов x и y. Вычисления контекстной близости для пары вершин проводится по графу ключевых слов. При этом высокие частоты вхождений слов x и y в различные наборы негативно влияют на уровень близости: частотные слова склонны иметь больше общих контекстов. Таким образом, возникает естественная идея нормировки близости на частоты встречаемости слов, для которых необходимо вычислить уровень близости. Кроме того, поскольку слова x, y и k могут все входить в один набор, то в таком случае связь слов x и y через слово k будет отражать скорее факт совместной встречаемости x и y в одном наборе, а не контекстную близость этой пары слов. Совместная встречаемость далеко не всегда влечет сильный уровень семантической близости. Характер этой связи частотности и смысловой схожести во многом зависит от сферы, в которой применяются ключевые слова, и от того, с какой целью пользователи системы эти ключевые используют. Например, ключевые слова для научной публикации редко содержат в себе синонимы, поскольку эти слова используются, чтобы дать читателю понять, о чем будет статья. Точные синонимы для слов в этом случае не несут никакой дополнительной информации о данной области. Вследствие этого высокая частота совместной встречаемости не ведет к семантической близости и должна пессимизировать значение формулы контекстной близости. Другим примером являются ключевые слова в социальных сетях. Пользователи используют ключевые слова к документам таким образом, чтобы этот документ было легче найти среди других документов системы. Поэтому использование

синонимов, переводов, транслитерации, различных способов написания ключевых слов помогает в поиске документа. Это однако не добавляет никакой смысловой информации непосредственно к описанию этого документа. Примеры различных наборов ключевых слов из разных областей будут даны в разделе "Тестовые испытания". Исходя из этих соображений предлагаются две представленные далее формулы для вычисления контекстной близости для пары ключевых слов.

Для пары ключевых слов i, j контекстная близость определяется по формулам:

$$C_+(i,\ j) = \frac{C(i,\ j)\log\left(1 + m(i,\ j)\right)}{f(i) + f(j)},$$

$$C_-(i,\ j) = \frac{C(i,\ j)}{\left(1 + m(i,\ j)\right)\left(f(i) + f(j)\right)},$$

где $C(i, j)$ — контекстная близость между i, j внутри графа; $m(i, j)$ — частота совместной встречаемости в наборах пары i, j; $f(i)$ — индивидуальная частота встречаемости слова i в наборах. В программной реализации алгоритмов частоты $f(i)$ и $m(i, j)$ для удобства сохраняются при построении графа ключевых слов в качестве дополнительной информации, соответственно, в вершинах и в ребрах графа.

Построение полного контекстного графа

Вершинами контекстного графа, как и в случае графа, описанного выше, являются ключевые слова. Ребро в таком графе свидетельствует о том, что пара ключевых слов является контекстно близкой. Чтобы определить, соединены ли два ключевых слова ребром, необходимо подсчитать контекстную близость по формулам, приведенным в предыдущем разделе. В общем случае для определения всех связей потребовалось бы $O(n^2)$ действий, где n — число вершин. Однако с помощью представленного далее алгоритма эта задача может быть решена за $O(nm^2)$ действий, где m — максимальное число соседей у вершины в графе ключевых слов.

Шаг 1. Построение графа ключевых слов G по входным наборам ключевых слов D.

Шаг 2. Подсчет частот встречаемости $f(i)$ и $m(p, q)$ для каждого слова i, пары слов (p, q) по входным наборам из D.

Шаг 3. Инициализация разреженной матрицы C размером $n \times n$.

Шаг 4. Для каждой вершины i графа ключевых слов:

a) для каждой пары соседей (p, q) вершины i:

$C(p, q)$ + = min($G(p, i)$, $G(q, i)$).

Шаг 5. Для каждой ненулевой пары (i, j) матрицы C:

a) $C_+(i,\ j) = \frac{C(i,\ j)\log\left(1 + m(i,\ j)\right)}{f(i) + f(j)}$;

b) $C_-(i,\ j) = \frac{C(i,\ j)}{\left(1 + m(i,\ j)\right)\left(f(i) + f(j)\right)}$.

Утверждение 1. Расчет весов $C_+(i, j)$ и $C_-(i, j)$ по построенному графу ключевых слов имеет сложность $O(nm^2)$, где n — число вершин графа ключевых слов; m — максимальное число ребер у вершины в графе ключевых слов.

Доказательство. Поскольку индивидуальные и парные частоты $f(i)$ и $m(i, j)$ уже были рассчитаны при построении графа ключевых слов, их получение имеет сложность $O(1)$. Остается лишь вычислить сложность подсчета формул $C(p, q)$. Для вычисления требуется пройтись по всем n вершинам графа и для каждой вершины рассмотреть все пары ее соседей. Поскольку у вершины не более m соседей, то обработка одной вершины занимает $O(m^2)$, а всех вершин, соответственно, $O(nm^2)$. Таким образом, общее время работы алгоритма составляет $O(nm^2)$, что и требовалось доказать.

В целях большей оптимизации времени на построение контекстного графа разумным является ограничение числа рассматриваемых соседей для текущей вершины i. Данная оптимизация выполняется следующей модификацией шага 4 описанного ранее алгоритма.

Шаг 4. Для каждой вершины i графа ключевых слов:

a) сортировка соседей вершины i по убыванию весов в ребрах;

b) выделение множества $N_k(i)$ — первых k соседей из сортированного на шаге 4, a списка соседей для вершины i;

c) для каждой пары соседей (p, q) из множества $N_k(i)$ вершины i:

$C(p, q)$ + = min($G(p, i)$, $G(q, i)$).

В данном случае появляется дополнительный параметр модели k (обычно в диапазоне от 10 до 30), который подбирается исходя из природы коллекции данных, а также по вычислительной производительности машины, на которой запущены расчеты. Отметим, что предварительная сортировка соседей вершины i по весам ребер позволяет использовать наиболее важных соседей в первую очередь. В результате этого на практике появляется способ значительно уменьшить количество вычислений и при этом построить модель, не уступающую в качестве модели, в которой рассматривается полный набор соседей для вершины. В некоторых случаях удаление менее значимых вершин дает даже прирост в качестве, поскольку зачастую такие вершины являются шумовыми для определения семантической близости.

Построение контекстного графа, в котором для вершины рассматриваются не все ее соседи, имеет вычислительную сложность $O(nk^2 + m\log(m))$, что показано в следующем утверждении.

Утверждение 2. Расчет весов $C_+(i, j)$ и $C_-(i, j)$ по построенному графу ключевых слов для случая ограниченного числа рассматриваемых соседей для текущей вершины имеет сложность $O(nk^2 + m\log(m))$, где n — число вершин графа ключевых слов, k — число рассматриваемых соседей.

Доказательство. Аналогично предыдущему утверждению, за исключением того, что теперь для текущей вершины будет рассмотрено порядка $O(k^2)$ пар соседей. Кроме того, появляются дополнительные затраты на сортировку соседей вершины (шаг 4, *а* модифицированного алгоритма), которые занимают $O(m\log(m))$ времени.

При $k = \sqrt{m}$, например, достигается оценка $O(nm)$ времени работы, что является значительным ускорением работы алгоритма.

Таким образом, данный алгоритм позволяет за время, относительно небольшое по сравнению с наивным перебором всех пар вершин, рассчитать контекстный граф, собранный по данным из миллионов наборов ключевых слов, поскольку в среднем каждое слово соединено с небольшим числом других слов. Следует также отметить, что отношение контекстной близости коммутативно, поэтому достаточно хранить только верхний правый угол матрицы $C(p, q)$.

Построение усеченного контекстного графа

Обозначим теперь за $C_*(i, j)$ любую из формул $C_+(i, j)$ или $C_-(i, j)$. Ненулевое значение $C_*(i, j)$ означает контекстную связь между ключевыми словами. Несмотря на то что такие матрицы контекстной близости остаются сильно разреженными, существует огромное число ненулевых связей, что делает затруднительным их дальнейший анализ с технической точки зрения. В дополнение к этому, низкие значения близости могут являться шумом. Такие данные не привносят полезной информации, а только ухудшают качество алгоритмов. По этим причинам возникает практическая необходимость в усечении графа, собранного описанным выше алгоритмом. Под усечением понимается удаление ребер, которые представляют наименее качественные и статистически проверенные связи.

По результатам экспериментов на коллекциях данных, описанных далее в разделе "Тестовые испытания", была установлена следующая стратегия отбора важных связей для вершины *i*.

1. Для всех *j* удаляются связи со слишком низким уровнем близости $C_*(i, j)$. Этот шаг необходим, чтобы удалить "шумные" и слабые связи из рассмотрения. Стоит также отметить, что в результате экспериментов было проверено, что одного этого правила недостаточно для качественного обрезания лишних ребер. Причиной этому является тот факт, что сами значения близости $C_*(i, j)$ не так важны, как порядок, который они задают на множестве соседей вершины *i*. Другими словами, данную задачу фильтрации стоит рассматривать как задачу ранжирования соседей вершины *i*, а не как задачу классификации пар (i, j) на классы полезных и бесполезных ребер.

2. От оставшихся выбирается некоторая доля связей (например, 20 %) с наибольшими значениями $C_*(i, j)$. Это условие представляется естественным, потому что слова, которые встречаются со многими другими в одинаковых контекстах, должны иметь больше ребер в графе, чем те слова, которые контекстно близки только с небольшим числом слов.

3. Число отобранных связей должно находиться в некоторых рамках (например, не менее трех и не более десяти соседей на вершину). Верхняя граница является преимущественно техническим, основанным на опыте автора, ограничением. Если было бы взято 20 % от числа всех соседей, но это число по-прежнему было бы достаточно велико, то хранение таких вершин потребовало бы значительных ресурсов. Нижняя граница берется чтобы вершина, для которой имеется мало кандидатов, получила хотя бы их в качестве ребер. С учетом ограничения из п. 1 можно ожидать, что эти связи будут достаточно качественными для дальнейшего анализа.

Следует также отметить, что окончательное число соседей для данной вершины *i* может быть несколько больше, поскольку лишние связи могли породиться одним из соседей вершины в полном графе, это означает, что ребро (i, j) может существовать, потому что оно прошло фильтрацию ребер для вершины *j*, а не для вершины *i*.

Далее приведено более формальное описание алгоритма, реализующего введенную выше модель.

Шаг 1. Все значения близости, меньшие порогового *t*, приравниваются нулю:

$$C_*(i, j) := 0, \text{ если } C_*(i, j) < t.$$

Шаг 2. Пусть $rank_j$ — порядковый номер соседа *j* в отсортированном по $C_*(i, j)$ списке всех соседей вершины *i*. Тогда, если $rank_j > \max(n_{\min}, \min(n_{\max}, n \cdot r))$, то связь (i, j) должна быть отфильтрована. Здесь $n_{\min}$, $n_{\max}$ — минимальное и максимальное число ребер для одной вершины в новом графе; *n* — число соседей вершины в полном контекстном графе; *r* — доля ребер, которую необходимо перенести в усеченный граф.

Среди описанных выше параметров только параметр *t* требует анализа для подбора. Остальные пороговые значения легко могут быть выбраны исходя из специфики задачи. Подбирать *t* можно эмпирически или по небольшой размеченной выборке примеров контекстно похожих слов. Отметим, что *t* должен быть выбран так, чтобы полнота выбранных ребер оставалась достаточно высокой.

Отметим также, что ребра построенного графа могут быть помечены числами, и представляется разумным введение функции расстояния между вершинами. Например, такой функцией может служить величина $C_*(i, j)$. В рамках рассматриваемого подхода ребра графа остаются непомеченными, что существенно понижает сложность разработанных моделей.

Процедура кластеризации усеченного контекстного графа

Ребра усеченного контекстного графа, который был введен в предыдущем разделе, в большей мере показывают семантическую близость между парой вершин, чем ребра полного контекстного графа или, тем более, графа ключевых слов. Помимо того что усеченный граф значительно уменьшает вычислительные

затраты, он повышает точность выявленных семантических связей между ключевыми словами, жертвуя при этом полнотой. В связи с этим рассмотрение длинных путей в графе становится более оправданным, поскольку семантическая близость между парой вершин лучше сохраняется с увеличением расстояния в графе. Вследствие этого возникает задача кластеризации графа: разделение множества всех вершин на подмножества таким образом, что любая пара вершин из одного множества является парой близких по смыслу ключевых слов.

За основу кластеризующего алгоритма взят алгоритм Louvain Modularity [30]. В ходе работы алгоритма максимизируется значение функционала модульности:

$$Q = \frac{1}{2m}\sum_{i,j}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_j, c_j),$$

где δ — дельта-функция; A_{ij} — вес ребра между вершинами i и j; $k_i = \sum_j A_{ij}$; $m = \frac{1}{2}\sum_{i,j} A_{ij}$. Как показано авторами работы [31], задача максимизации описанного выше функционала является NP-сложной, поэтому для ее решения используется аппроксимационный алгоритм, основные шаги которого описаны далее.

Шаг 1. Для каждой вершины графа создается свой кластер.

Шаг 2. Для каждой вершины i и для каждого соседа j вершины i:

a) временное добавление вершины i в кластер вершины j;

b) подсчет изменения оптимизируемого функционала Q;

c) окончательное добавление вершины i в кластер того соседа, на котором достигается максимальное увеличение значения Q; если функционал невозможно увеличить, то добавления не происходит.

Шаг 3. Построение нового графа, вершинами которого являются кластеры, а веса ребер отражают связи между кластерами. Вес ребра равен сумме весов всех пар ребер, вершины которых лежат в соответствующих кластерах.

Преимущество описанного алгоритма в его масштабируемости на графы больших размеров. Качество кластеризации при этом остается на высоком уровне. Отметим, что число кластеров не является параметром данного алгоритма. На практике кластеры, получающиеся в результате работы программной реализации алгоритма, оказываются слишком большого размера. В некоторые кластеры могут попасть тысячи или десятки тысяч слов, очевидно, что не существует такого огромного множества попарно похожих по смыслу слов. Алгоритм является общим графовым алгоритмом и никаким образом не использует информацию о семантической близости. Даже точное решение оптимизационной задачи не гарантирует качественного разбиения вершин графа на подмножества, элементы которого семантически близки друг к другу. Как следствие, необходимы дополнительные действия, связывающие процессы кластеризации графа и определения семантической близости. Далее представлена окончательная версия алгоритма кластеризации контекстного графа, разрешающая отмеченную трудность.

Шаг 1. Заводится очередь для подграфов исходного графа, исходный граф добавляется в нее.

Шаг 2. Пока очередь не пуста:

a) кластеризация подграфа из очереди алгоритмом Louvain Modularity;

b) для каждого полученного в результате кластеризации подграфа-кластера:

i) если размер кластера меньше, чем k, то добавить кластер в выходное множество кластеров;

ii) иначе добавить кластер в очередь подграфов.

Параметр алгоритма k выбирается из специфики задачи. Для задачи кластеризации ключевых слов значение параметра k может варьироваться в пределах от 10 до 20.

Тестовые испытания

В настоящем разделе описаны результаты тестовых испытаний программных реализаций описанных выше алгоритмов. В качестве тестовых данных были использованы корпусы ключевых слов для научных публикаций, собранных в сети Интернет. Использовалась также информация из социальной сети ВКонтакте, а именно были выкачаны посты (публичные сообщения из групп и страниц пользователей), часть из которых помечена хэштегами. В процессе сбора данных проводился парсинг текстовых данных на предмет наличия в них наборов ключевых слов. Точное решение этой задачи не является предметом исследования настоящей работы, поэтому для парсинга данных были использованы наивные подходы, которые, тем не менее, позволяют собрать корпус достаточного размера и качества для проведения дальнейшего анализа.

В конечном итоге собрано два объемных набора данных:

1) 329 000 наборов для русского языка;

2) 3 069 000 наборов хэштегов из сети ВКонтакте.

Далее приведены примеры наборов ключевых слов из обоих источников.

Наборы ключевых слов научных публикаций:

[топонимический концепт, языковое сознание, когнитивная база, прецедентность, апеллятивация];

[вариабельность сердечного ритма, гребля на каноэ, вегетативный тонус];

[архитектуры, деформации, геологическая среда, сфера взаимодействия].

Наборы хэштегов из социальной сети:

[electro_pop, dance, fresh, music, new_zealand];

[vitaminhealth, oxygenwater, waterhealth];

[bodyfan, питание, bodyfanпитание, bodyfanmotivation, motivation, bodybuilding, фитнес, gym, спорт, мотивация, зож].

Программные реализации описанных в предыдущих разделах алгоритмов были применены к собранным данным. На рис. 1 и 2 представлены ближайшие

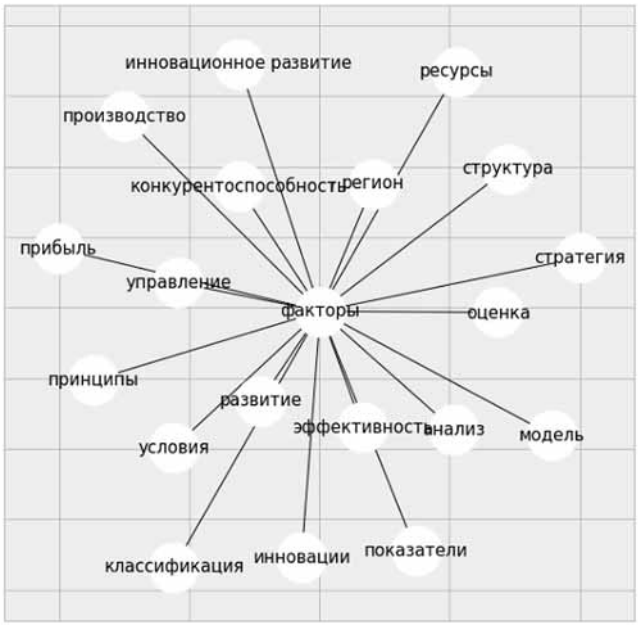


Рис. 1. Соседи вершины "факторы" в графе ключевых слов

Рис. 2. Соседи вершины "регионы" в графе ключевых слов

соседи для слов "федерация" и "регионы" в графе ключевых слов.

На рис. 1 и 2 видно, что графа ключевых слов не хватает для определения семантической близости пары слов — существуют связи такие, как "факторы — модель", "регионы — экономика", которые не обладают явной смысловой связью. Применение методов построения усеченного контекстного графа дает значительное улучшение качества классификации пар ключевых слов на семантически близкие и далекие.

Далее указаны примеры найденных пар ключевых слов, близких по смыслу:

β-адреноблокаторы — бета-адреноблокаторы новые виды — новый вид орви — острые респираторные вирусные инфекции;

текущий уровень информационной безопасности — политика информационной безопасности умения — навыки;

образное мышление — художественный вкус;

хехцир — khekhtsyr;

рынок банковских услуг — банковский рынок;

тромболизис — тромболитическая терапия;

параллельные алгоритмы — параллельное программирование;

феминность — фемининность;

полином — многочлен;

корень — корни;

primerun — примерун fvk;

fotovideoclub еврореволюция — єврореволюція;

silk_plaster — шелковая_штукатурка.

Интересным фактом является определение похожих слов для заданного многозначного слова. В то время как граф в графе ключевых слов соседями для слова "орган" являются слова "государство", "сибирь", "контроль", "циркуляция", "управление", в контекстном графе ближайшими являются слова "музыковедение", "организм", "отклонение", "делегирование полномочий", "объект контроля". Таким образом, восстанавливается не только значение слова, связанное с юриспруденцией, но и близкие слова для значения из области музыки ("музыковедение") и биологии ("организм").

Кластеризация контекстного графа позволяет удалять недостаточно надежные связи между словами и, наоборот, добавлять новые ребра между семантически похожими парами слов. Например, на рис. 3

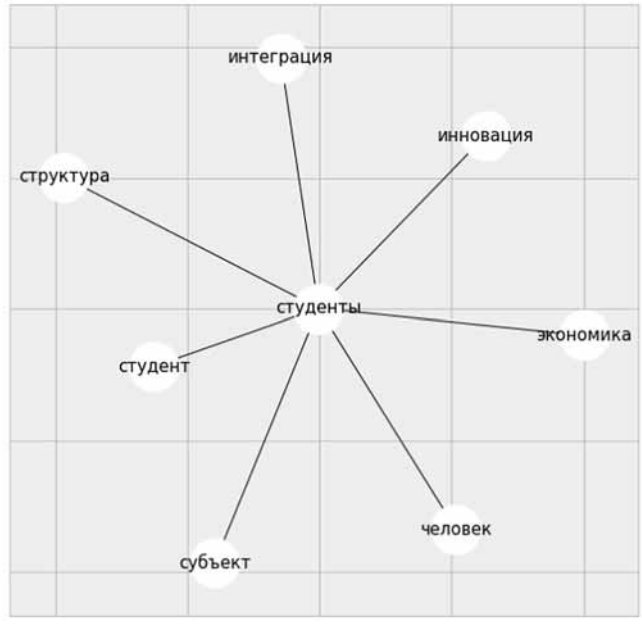


Рис. 3. Наиболее близкие слова для слова "студенты" в контекстном графе

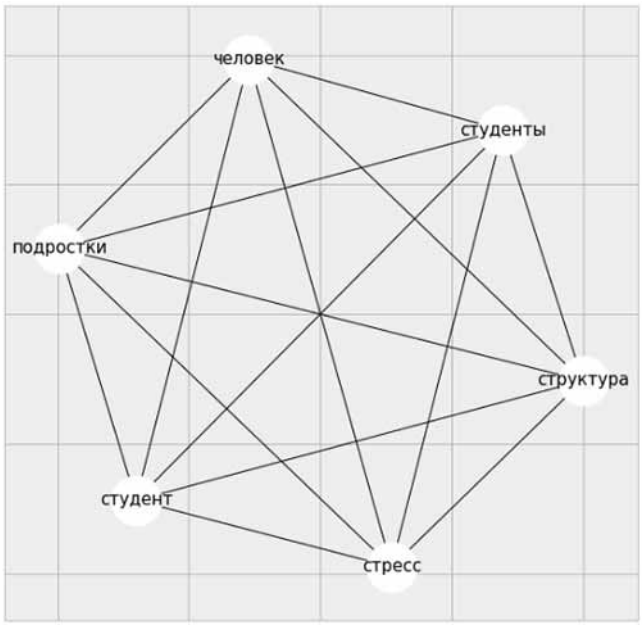


Рис. 4. Кластер, содержащий слово "студенты"

изображены несколько ближайших контекстно близких слов для слова "студенты" (отметим, что полное множество соседей вершины слишком велико, чтобы его изобразить).

Кластер для слова "студенты" изображен на рис. 4.

Можно заметить, как в результате кластеризации были разорваны связи "студенты — экономика", "студенты — инновации", вместо которых на первый план вышли связи "студенты — подростки". Отметим также, что выбранный метод кластеризации графа может допускать ошибки. Например, в случае с парой "студенты — структура", которая была как в усеченном контекстном графе, так и в кластере слова "студент".

Для интегральной проверки качества была выбрана следующая методика.

1. Набирается набор пар ключевых слов, для которых можно определить их высокий уровень семантической близости посредством детерминированного алгоритма (положительные примеры). Пара слов определяется близкой по смыслу, если выполняется хотя бы одно из условий:

a) одно ключевое слово является аббревиатурой для другого;

b) расстояние Левенштейна между парой невелико.

2. К зафиксированным семантически похожим парам слов добавляются отрицательные примеры, т. е. пары слов, которые не являются близкими по смыслу. Для этого проводятся следующие шаги:

a) если пара слов $< a, b >$ определена на первом шаге как пара похожих слов, то для слова a берется k случайных соседей $c_1, c_2, ..., c_k$ из графа ключевых слов на расстоянии, не превышающем 2;

b) все пары $< a, c_i > \; i \in [1, k]$ определяются как отрицательные примеры.

3. Положительные и отрицательные примеры составляют тестовую выборку. После чего для всех пар тестовой выборки вычисляется близость, по формулам, описанным выше, а также подбирается порог, по которому в зависимости от вычисленного значения близости пара относится либо к классу семантически близких пар, либо к классу семантически далеких пар.

4. По оценкам классификатора и тестовой выборке считается F-мера, которая и является показателем качества алгоритма.

В результате тестирования программной реализации алгоритма получено высокое по отношению к реализованным в работе [25] алгоритмам значение F-меры — 0,82. Проведение аналогичного теста для алгоритма определения близости лишь с помощью анализа частотности встречаемости пары слов дает результат 0,67, таким образом, методы, описанные в данной работе, существенно улучшают качество определения семантической похожести. Отметим, что выбранный способ тестирования имеет очевидный недостаток: среди положительных примеров очень редко встречаются пары смысловых синонимов, напротив, они могут попадать в пары отрицательных примеров. Тем не менее по экспертной оценке увеличение качества на описанной ранее тестовой выборке влечет улучшение качества классификации и более сложных пар ключевых слов, таких как "синоним — синоним" или "слово — перевод слова на другой язык".

Заключение

По результатам исследований, результаты которых представлены в работе, построены модели определения близости по корпусу наборов ключевых слов, опирающиеся на методы из теории графов. Для данных моделей представлены алгоритмы и созданы программные реализации этих алгоритмов. Реализации были протестированы на двух коллекциях наборов ключевых слов, и был получен относительно высокий уровень качества результатов. Кроме того, была разработана модель кластеризации ключевых слов, опирающаяся на введенные графовые модели представления коллекций ключевых слов и на построенную по этим графам меру схожести для пары слов. Программная реализация процедуры кластеризации также протестирована, для нее был получен высокий уровень качества определения кластеров схожих ключевых слов.

Недостатком работы может являться большое число параметров, которое необходимо настроить. В качестве дальнейшего направления в изучении данной области автор настоящей публикации считает целесообразным применение техник машинного обучения для построения меры близости между ключевыми словами. Такой шаг поможет избавиться от значительной части параметров моделей, описанных в статье, предоставив возможность настройки этих параметров в автоматическом режиме.

Автор выражает благодарность д-ру физ.-мат. наук., проф. В. А. Васенину и канд. физ.-мат. наук, вед. науч. сотр. С. А. Афонину за внимание к работе и помощь в подготовке статьи.

Список литературы

1. **Levenshtein V.** Binary Codes Capable of Correcting Deletions, Insertions and Reversals // Soviet Physics Doklady. 1966. Vol. 10. P. 707.
2. **Miller F. P., Vandome A. F., McBrewster J.** Levenshtein Distance: Information Theory, Computer Science, String (Computer Science), String Metric, Damerau-Levenshtein Distance, Spell Checker, Hamming Distance, Alpha Press, 2009.
3. **Jacobs J.** Finding words that sound alike. The SOUNDEX algorithm // Byte 7. 1982. P. 473—474.
4. **Hixon B., Schneider E., Epstein S. L.** Phonemic Similarity Metrics to Compare Pronunciation Methods // INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, August 27-31, 2011. Florence, Italy. 2001. P. 825—828. URL: http://www.isca-speech.org/archive/interspeech_2011/i11_0825.html
5. **Albatineh A. N., Niewiadomska-Bugaj M.** Correcting Jaccard and other similarity indices for chance agreement in cluster analysis // Advances in Data Analysis and Classification, 2011. Vol. 5, No. 3. P. 179—200. DOI: 10.1007/s11634-011-0090-y.
6. **Ristad E. S., Yianilos P. N., Member S.** Learning string edit distance // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998. Vol. 20. P. 522—532.
7. **Church K. W., Hanks P.** Word Association Norms, Mutual Information, and Lexicography // Comput. Linguist. 1990. Vol. 16, No. 1. P. 22—29, URL: http://dl.acm.org/citation.cfm?id=89086.89095.
8. **Dunning T.** Accurate Methods for the Statistics of Surprise and Coincidence//Comput. Linguist. 1993. Vol. 19, No. 1. P. 61—74.
9. **Chen S., Goodman J.** An Empirical Study of Smoothing Techniques for Language Modeling // Proceedings of the 34th Annual Meeting on Association for Computational Linguistics. 1996. P. 310—318.
10. **Rosenfeld R.** Adaptive Statistical Language Modeling: a Maximum Entropy Approach, School of Computer Science, Carnegie Mellon University, 1994. P. 94—138.
11. **Schneider K.-M.** Weighted Average Pointwise Mutual Information for Feature Selection in Text Categorization// Knowledge Discovery in Databases: PKDD 2005. P. 252—263.
12. **Dagan I., Lee L., Pereira F. C. N.** Similarity-Based Models of Word Cooccurrence Probabilities // Machine Learning. 1999. Vol. 34, No. 1. P. 43—69.
13. **Bouma G.** Normalized (pointwise) mutual information in collocation extraction // From Form to Meaning: Processing Texts Automatically, Proceedings of the Biennial GSCL Conference 2009. 2009. P. 31—40.
14. **Thanopoulos A., Fakotakis N., Kokkinakis G.** Comparative Evaluation of Collocation Extraction Metrics // Proceedings of the Third International Conference on Language Resources and Evaluation. 2002. P. 620—625.
15. **Miller G. A.** WordNet: A Lexical Database for English // Communications of the ACM. 1995. Vol. 35, No. 11. P. 39—41.
16. **Braslavski P., Ustalov D., Mukhin M.** A Spinning Wheel for YARN: User Interface for a Crowdsourced Thesaurus // Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics. 2004. P. 101—104.
17. **Braslavski P.** YARN: Spinning-in-progress // Proceedings of the 8th Global WordNet Conference. 2016. P. 58—65.
18. **Azarova I.** RussNet: Building a Lexical Database for the Russian Language // Workshop on Wordnet Structures and Standardisation and How this affect Wordnet Applications and Evaluation. 2002. P. 60—64.
19. **Лукашевич Н.** Тезаурусы в задачах информационного поиска. М.: Изд-во Моск. ун-та, 2011. 387 с.
20. **Loukachevitch N. V., Lashevich G., Gerasimova A. A., Ivanov V. V., Dobrov B. V.** Creating Russian WordNet by Conversion // Komputernaya lingvistika i intellektualnie tehnologii. Rossiiskiy gosudarstvenniy gumanitarniy universitet. 2016. P. 405—415.
21. **Gabrilovich E., Markovitch S.** Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis // Proceedings of the 20-th International Joint Conference on Artifical Intelligence. 2007. P. 1606—1611.
22. **Turdakov D., Velikhov P.** Semantic Relatedness Metric for Wikipedia Concepts Based on Link Analysis and its Application to Word Sense Disambiguation // CEUR Workshop Proceedings. 2008. P. 355.
23. **Mikolov T.** Distributed Representations of Words and Phrases and Their Compositionality // Proceedings of the 26th International Conference on Neural Information Processing Systems, 2013. P. 3111—3119.
24. **Kenter T., de Rijke M.** Short Text Similarity with Word Embeddings // Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. 2015. P. 1411—1420.
25. **Афонин С. А., Лунев К. В.** Выявление тематических направлений в коллекции наборов ключевых слов // Программная инженерия. 2015. № 2. С. 29—39.
26. **Rafi M., Amin F., Shaikh M. S.** Document clustering using graph based document representation, Computing Research Repository (CoRR), 2014, URL: https://dblp.org/rec/bib/journals/corr/RafiAS14
27. **Stanchev L.** Fine-Tuning an Algorithm for Semantic Document Clustering Using a Similarity Graph // International Journal of Semantic Computing. 2016. Vol. 10. P. 527—555.
28. **Sedoc J., Gallier J., Ungar L., Foster D.** Semantic Word Clusters Using Signed Normalized Graph Cuts, Computing Research Repository (CoRR), 2016, available at: https://arxiv.org/pdf/1601.05403.pdf
29. **Wang P., Xu J., Xu B., Liu C., Zhang H., Wang F., Hao H.** Semantic Clustering and Convolutional Neural Network for Short Text Categorization // Annual Meeting of the Association for Computational Linguistics (ACL). 2015. P. 352—357.
30. **Blondel V., Guillaume J.-L., Lambiotte R., Lefebvre E.** Fast unfolding of communities in large networks // Journal of Statistical Mechanics: Theory and Experiment, IOP Publishing. 2008. P10008. P. 1—12.
31. **Brandes U., Delling D., Gaertler M. et al.** Maximizing Modularity is hard, arXiv Digital Library, 2006, available at: https://arxiv.org/pdf/physics/0608255

Graph Methods for Computing Semantic Similarity of a Pair of Keywords and Their Application to the Problem of Keywords Clustering

K. V. Lunev, e-mail: kirilllunev@gmail.com, Faculty of mechanics and mathematics, Lomonosov Moscow State University, Institute of mechanics, Moscow, 119192, Russian Federation

Corresponding author:

Lunev Kirill V., Postgraduate Student, Faculty of mechanics and mathematics, Lomonosov Moscow State University, Institute of mechanics, Moscow, 119192, Russian Federation,
E-mail: kirilllunev@gmail.com

Received on March 03, 2018
Accepted on April 13, 2018

The article presents the results of research on the direction of search models, algorithms and software to determine the semantic similarity between two keywords. The methods which are used in the work are based on the graph theory algorithms. The document is represented as a set of keywords associated with the document. A measure of contextual similarity of a pair of keywords is developed. A keywords graph is constructed for a given collection of documents. The nodes of the graph correspond to the keywords, and edges represent the fact of the contextual closeness of a pair of words. The method of clustering of the constructed graph is presented below. The keywords included in one cluster have the property of semantic similarity, which is an important result of this work. Software implementation of the developed models has been tested on the collections of scientific publications keywords, as well as on the collection of posts tags in the VKontakte social network.

Keywords: *semantic similarity, natural language processing, graph algorithms, graph theory, clustering*

For citation:

Lunev K. V. Graph Methods for Computing Semantic Similarity of a Pair of Keywords and Their Application to the Problem of Keywords Clustering, *Programmnaya Ingeneria*, 2018, vol. 9, no. 6, pp. 262—271.

DOI: 10.17587/prin.9.262-271

References

1. **Levenshtein V.** Binary Codes Capable of Correcting Deletions, Insertions and Reversals, *Soviet Physics Doklady*, 1966, vol. 10, pp. 707.

2. **Miller F. P., Vandome A. F., McBrewster J.** *Levenshtein Distance: Information Theory, Computer Science, String (Computer Science), String Metric, Damerau-Levenshtein Distance, Spell Checker, Hamming Distance*, Alpha Press, 2009.

3. **Jacobs J.** Finding words that sound alike. The SOUNDEX algorithm, *Byte 7*, 1982, pp. 473—474.

4. **Hixon B., Schneider E., Epstein S. L.** Phonemic Similarity Metrics to Compare Pronunciation Methods, *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy, August 27—31, 2011,* pp. 825—828, available at: http://www.isca-speech.org/archive/interspeech_2011/i11_0825.html

5. **Albatineh A. N., Niewiadomska-Bugaj M.** Correcting Jaccard and other similarity indices for chance agreement in cluster analysis, *Advances in Data Analysis and Classification*, 2011, vol. 5, no. 3, pp. 179—200. DOI: 10.1007/s11634-011-0090-y.

6. **Ristad E. S., Yianilos P. N., Member S.** Learning string edit distance, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998, vol. 20, pp. 522—532.

7. **Church K. W., Hanks P.** Word Association Norms, Mutual Information, and Lexicography, *Comput. Linguist.*, 1990, vol. 16, no. 1, pp. 22—29, available at: http://dl.acm.org/citation.cfm?id=89086.89095.

8. **Dunning T.** Accurate Methods for the Statistics of Surprise and Coincidence, *Comput. Linguist*, 1993, vol. 19, no. 1, pp. 61—74.

9. **Chen S., Goodman J.** An Empirical Study of Smoothing Techniques for Language Modeling, *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics,* 1996, pp. 310—318.

10. **Rosenfeld R.** Adaptive Statistical Language Modeling: a Maximum Entropy Approach, *School of Computer Science, Carnegie Mellon University,* 1994, pp. 94—138.

11. **Schneider K.-M.** Weighted Average Pointwise Mutual Information for Feature Selection in Text Categorization, *Knowledge Discovery in Databases: PKDD 2005,* pp. 252—263.

12. **Dagan I., Lee L., Pereira F. C. N.** Similarity-Based Models of Word Cooccurrence Probabilities, *Machine Learning,* 1999, vol. 34, no. 1 pp. 43—69.

13. **Bouma G.** Normalized (pointwise) mutual information in collocation extraction, *From Form to Meaning: Processing Texts Automatically, Proceedings of the Biennial GSCL Conference 2009*, 2009, pp. 31—40.

14. **Thanopoulos A., Fakotakis N., Kokkinakis G.** Comparative Evaluation of Collocation Extraction Metrics, *Proceedings of the Third International Conference on Language Resources and Evaluation,* 2002, pp. 620—625.

15. **Miller G. A.** WordNet: A Lexical Database for English, *Communications of the ACM*, 1995, vol. 35, no. 11, pp. 39—41.

16. **Braslavski P., Ustalov D., Mukhin M.** A Spinning Wheel for YARN: User Interface for a Crowdsourced Thesaurus, *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics,* 2004, pp. 101—104.

17. **Braslavski P.** YARN: Spinning-in-progress, Proceedings of the 8th Global WordNet Conference, 2016, pp. 58—65.

18. **Azarova I.** RussNet: Building a Lexical Database for the Russian Language, *Workshop on Wordnet Structures and Standardisation and How this affect Wordnet Applications and Evaluation*, 2002, pp. 60—64.

19. **Loukachevitch N. V.** *Tezaurusi v zadachah informacionnogo poiska* (Thesaurus in information retrieval problems), Moscow, Izdatelstvo Moskovskogo Universiteta, 2011, 387 p. (in Russian).

20. **Loukachevitch N. V., Lashevich G., Gerasimova A. A., Ivanov V. V., Dobrov B. V.** Creating Russian WordNet by Conversion, *Komputernaya lingvistika i intellektualnie tehnologii,* Rossiiskiy gosudarstvenniy gumanitarniy universitet, 2016, pp. 405—415.

21. **Gabrilovich E., Markovitch S.** Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis, *Proceedings of the 20-th International Joint Conference on Artifical Intelligence*, 2007, pp. 1606—1611.

22. **Turdakov D.** Semantic Relatedness Metric for Wikipedia Concepts Based on Link Analysis and its Application to Word Sense Disambiguation, *CEUR Workshop Proceedings*, 2008, pp. 355.

23. **Mikolov T.** Distributed Representations of Words and Phrases and Their Compositionality, *Proceedings of the 26th International Conference on Neural Information Processing Systems,* 2013, pp. 3111—3119.

24. **Kenter T.** Short Text Similarity with Word Embeddings, *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 2015, pp. 1411—1420.

25. **Afonin S. A., Lunev K. V.** Viyavlenie tematicheskih napravleniy v kollekcii naborov kluchevih slov (Topic Analysis in Collection of Keyword Tuples), *Programmnaya Ingeneria*, 2015, vol. 2, pp. 29—39 (in Russian).

26. **Rafi M., Amin F., Shaikh M. S.** Document clustering using graph based document representation, *Computing Research Repository (CoRR),* 2014, available at: https://dblp.org/rec/bib/journals/corr/RafiAS14

27. **Stanchev L.** Fine-Tuning an Algorithm for Semantic Document Clustering Using a Similarity Graph, *International Journal of Semantic Computing*, 2016, vol. 10, pp. 527—555.

28. **Sedoc J., Gallier J., Ungar L., Foster D.** Semantic Word Clusters Using Signed Normalized Graph Cuts, *Computing Research Repository (CoRR),* 2016, available at: https://arxiv.org/pdf/1601.05403.pdf

29. **Wang P., Xu J., Xu B., Liu C., Zhang H., Wang F., Hao H.** Semantic Clustering and Convolutional Neural Network for Short Text Categorization, *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015, pp. 352—357.

30. **Blondel V., Guillaume J.-L., Lambiotte R., Lefebvre E.** Fast unfolding of communities in large networks, *Journal of Statistical Mechanics: Theory and Experiment*, 2008, P10008, pp. 1—12.

31. **Brandes U., Delling D., Gaertler M., Goerke R., Hoefer M., Nikoloski Z., Wagner D.** *Maximizing Modularity is hard*, arXiv Digital Library, 2006, available at: https://arxiv.org/pdf/physics/0608255

УДК 004.942 DOI: 10.17587/prin.9.272-280

А. А. Артемов, инженер, соискатель, e-mail: artemsince2@ya.ru, АНО Центр проблем стратегических ядерных сил Академии военных наук, г. Юбилейный

Предиктивная оценка верхней границы ошибки прогноза модели, возникающей вследствие концептуального смещения данных на примере мем-грамм-модели

Представлено решение задачи, направленной на оценку ошибки прогноза статистической вероятности события — наличия набора признаков у объектов модели заданной предметной области. Автор рассматривает комбинацию признаков объекта в виде n-грамм. Объектом модели предметной области является уникальная n-грамма фиксированной длины. Совокупность "мутирующих" объектов, объединенных по заданному критерию (например, время), образуют эволюционирующие мультимножества вариативной мощности. При таком формальном описании модели, фактором, идентифицирующим присутствие набора признаков, является наличие общего элемента двух мультимножеств — мем-граммы, а вероятность данного события определяется функционалом от числа копий мем-граммы. Модель, в которой определена данная аксиоматика, именуется автором мем-грамм-моделью.

Предложенное решение акцентирует внимание на рассмотрении вопросов оценки статистической вероятности повторяющихся элементов мультимножеств при условии возможности прогноза их численности только для части этих элементов. Такое решение востребовано в процессе разработки моделей представления знаний для самообучаемых интеллектуальных систем в условиях ограниченного объема обучающих примеров из общего объема перманентно изменяющихся больших данных.

Ключевые слова: *мем-грамм-модель, смещение данных, элементы наследственности, меметический алгоритм, эволюционирующая система, подобные мультимножества, вложенное мультимножество, теорема Расторгуева*

Введение

В последние годы все шире применяется практика машинного обучения при исследовании моделей знаний (далее для краткости изложения — "модели") на основе анализа больших данных. В классе интеллектуальных систем, использующих возможности машинного обучения, особый интерес представляют самообучаемые интеллектуальные системы (СИС). Любая СИС не только выполняет типовые задачи машинного обучения, например классификации или прогнозирования [1], но и демонстрирует способность к обучению в процессе работы. Адекватное функционирование таких систем невозможно без ответа на вопрос: "Насколько сильно изменилось знание с момента последнего обучения системы?" или, что равнозначно, "Как часто надо переобучать модель знаний?". Ответом на этот вопрос является возможность получения оценки ошибки прогноза на модели в момент его формирования.

Очевидно, что в случае динамической природы формирования данных, характеризующих модель, актуальность накопленных примеров обратно пропорциональна скорости изменения данных. В то же время большие данные — это лишь фрагмент еще бóльших данных, и поэтому всегда найдутся обстоятельства прагматического характера, требующие ограничить объем данных для обучения [2]. Понятно и то обстоятельство, что, если элементы больших данных задают соответствие между признаковым и объектным пространством и сочетание признаков объекта нельзя рассматривать как случайное независимое событие, то использование Центральной предельной теоремы и ее вариантов для данных элементов некорректно [3]. В науке о данных проблема получила название "смещение данных". Суть ее заключается в различиях между набором данных, который использовался для обучения модели, и набором данных, по которому проводится верификация модели. В отечественной научной литературе этот вопрос не получил должного освещения. В то же время в зарубежных научных публикациях уже можно найти несколько вариантов наименования проблемы: "Data fracture", "Dataset

Shift", "Changing environment", "Concept shift", "Changes of classification" [4].

Среди основных причин появления эффекта смещения данных определяют две — некорректную выборку данных для обучения и нестационарную среду, генерирующую данные [5]. Выделяют также следующие три базовых типа смещения данных:

а) сдвиг переменных (*covariative shift*) — изменения во входных данных или признаках;

б) смещение априорной вероятности (*prior probability shift*) — изменения в целевом показателе или классе;

в) концептуальное смещение (*concept shift*) — изменения в отношении между независимыми переменными и целевым показателем.

Идентифицируют наличие смещения данных с использованием специальных методов и подходов. К их числу относят следующие:

1) отслеживание соответствий (*correspondence tracing*) [6], подразумевается наличие классификатора, основанного на правилах: для каждой единицы данных идентифицируют и сравнивают старые и новые правила классификации;

2) концептуальная эквивалентность (*conceptual equivalence*) [7], при использовании которой подразумевается анализ данных напрямую, без сравнения правил классификации;

3) статистический метод (*a statistical framework*) [8], применение которого подразумевает расчет и сравнение прогнозов для каждого набора данных.

Ряд исследователей акцентируют внимание на способах преодоления эффекта смещения данных [9], большинство из которых направлены на трансформацию "нового" набора данных для улучшения качества прогноза модели, построенной на "старом" наборе данных. Другие исследователи предлагают менять саму модель так, чтобы минимизировалась средняя ошибка получаемых с помощью модели прогнозов на основании данных обучающей и тестовых выборок.

Однако вопрос, на который обращает внимание автор работы, лежит в другой плоскости. Существует ли возможность заранее определить (а значит, и учесть) ошибку модели, связанную с изменением данных об эволюционирующей среде, основываясь лишь на истории изменения данных? Так как в литературе по данной тематике сходных исследований обнаружено не было, в настоящей работе будет предложен способ решения представленной задачи при использовании программной реализации вероятностной модели языка на базе *n*-грамм [10—12].

Решение приведенной задачи потребовалось автору в процессе апробации разработанной им мем-грамм-модели. Это дискретная конечная математическая модель, которая является вариацией вероятностной языковой модели на базе *n*-грамм. В ней выбираются не все *n*-граммы, а только мемы — *n*-граммы, обладающие лучшими характеристиками для "выживания", позволяющими им реплицировать свои копии во времени. В терминах мем-грамм-модели изменяющиеся большие данные представлены последовательностью мультимножеств, а примеры — соответствующими им подмультимножествами.

Неформальная постановка задачи в рамках мем-грамм-модели может быть проиллюстрирована следующим описанием. Известно некоторое число страниц текста. Содержание каждой страницы представлено мультимножеством словосочетаний. На последовательности таких мультимножеств обучена мем-грамм-модель. Данная модель предсказывает, какие словосочетания предыдущей страницы будут находиться на следующей странице, а также значение относительной частоты для каждого из этих словосочетаний. Известно, что модель работает с высокой точностью, если максимальное число словосочетаний в содержании страниц соответствует заданному критерию модели. Задачи для решения: оценить ошибку в прогнозах модели, если для прогноза будет использовано не все содержание предыдущей страницы, а только его часть; какую долю словосочетаний необходимо использовать, чтобы ошибка была не больше заданного уровня; как изменится ошибка, если максимальное число словосочетаний на странице непостоянно?

Задачи подобного формата возникают не только в лингвистике, но и в медицине, генетике, кибернетике, исторической антропологии и многих других областях. В общем случае это такие задачи, где процесс изменения данных может быть хорошо описан эволюционными алгоритмами, но сбор "свежего" исследовательского материала в масштабе больших данных вызывает затруднение. Вследствие такого затруднения появляется эффект концептуального смещения верифицирующих "небольших" данных относительно больших данных, используемых для обучения модели.

Формальная постановка задачи

Автор столкнулся с трудностями сбора данных для часто обновляемой СИС, использующей мем-грамм-модель [13]. С применением мем-грамм-модели описывается эволюционирующее во времени информационное пространство (ИП). Иллюстративно это описание представлено на рис. 1.

Формально ИП представляет изменяющуюся во времени вероятностную модель языка. В таких моделях определено конечное мультимножество элементов — словосочетаний. Вероятность появления элемента определяется числом контекстов — последовательности слов, в которых присутствует фрагмент словосочетания. Базовая *n*-грамм-модель для биграмм представляется в виде

$$P\left(w_n \mid w_{n-1}\right)=\frac{N\left(w_{n-1}\, w_n\right)}{\sum\limits_k N\left(w_{n-1} w_n^k\right)},$$

где $P\left(w_n \mid w_{n-1}\right)$ — относительная частота *n*-граммы, интерпретируемая как вероятность события — следования слова w_n после слова w_{n-1}; N — число встреч последовательности слов w_{n-1} и w_n^k в исследуемых текстах; k — индекс слова w_n, встречаемого после слова w_{n-1}.

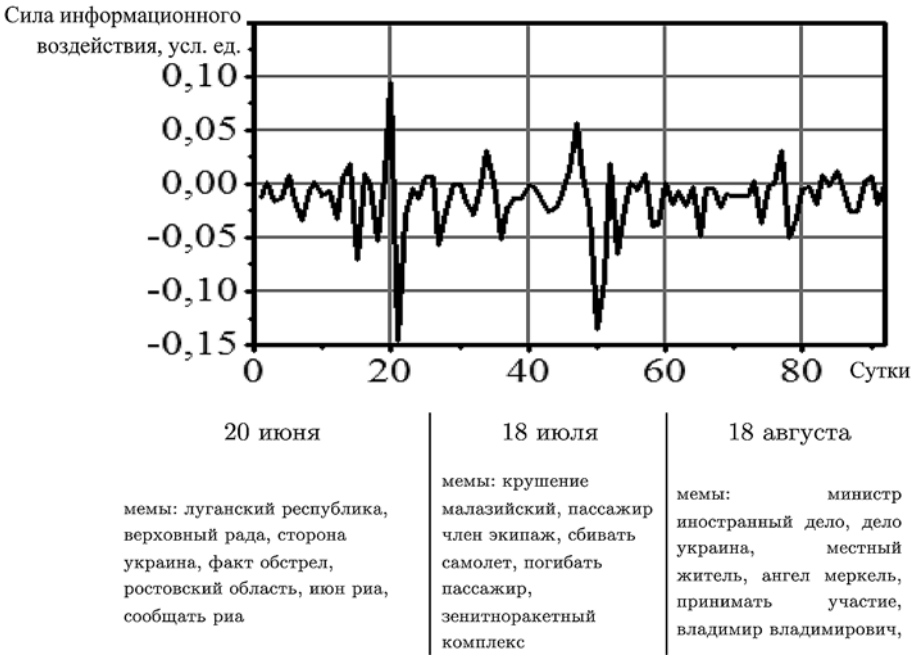


Рис. 1. Иллюстративное представление описания эволюции ИП с применением мем-грамм-модели: под графиком указаны даты, соответствующие отсчету времени (сут.) от выбранного начального момента времени (1 июня)

Наиболее удобной формой представления исходных данных о содержании ИП для машинного обучения являются мультимножества. Это обусловлено следующими обстоятельствами. Мультимножества объединяют возможности категориального и числового описания объектов [14]. Кроме того, от мультимножеств можно легко перейти к любой иной форме представления признакового пространства для заданных классов объектов — скалярной, векторной, матричной или тензорной [15].

Каждый этап эволюции ИП предполагает, что часть элементов мультимножества (мемов) остается неизменной (рис. 2, см. третью сторону обложки). Неизменная часть мультимножества характеризуется наследственностью, а противоположная ей — изменчивостью. Вместе эти две части образуют единое целое мультимножество.

Наибольший интерес для прогноза представляет неизменная (наследуемая) часть мультимножества. Учитывая это обстоятельство и тот факт, что доля мемов однозначно определяет возможности для оценки изменчивости ИП, автор разработал мем-грамм-модель для отбора и прогноза числа мемов. Эта модель состоит из двух частей: меметического алгоритма (двойной эволюционный алгоритм) отбора мемов [16] и модели прогноза числа копий мемов при условии заданного размера ИП [17]. Ниже представлен обобщенный алгоритм мем-грамм-модели.

Алгоритм мем-грамм-модели (обобщенный)

1. ***Вход1****: размер эталонного ИП, число копий n-граммы за 3 интервала времени, размер реального ИП. Максимальный размер наследственности M;*
1.1. Определение нормировочного коэффициента;
1.2. Отбор мемов — элементов наследственности множества M, представленных n-граммами, обладающих лучшей "приспособленностью" (fitness) в рассматриваемой популяции n-грамм;
While StopCondition *не выполнено* ***do***
1.2.1. Выбор родителей — слов n-граммы;
1.2.2. Получение представителей "потомства", похожих на выбранную n-грамму (к примеру, по первому слову);
1.2.3. Улучшение потомства, обычно путем локального поиска — отбираем n-граммы с максимальным критерием "приспособленность" (предлагается следующая мера — произведение числа копии на ускорение изменения относительной частоты за три периода);
1.2.4. Популяция обновляется в соответствии с правилом разнообразия (к примеру, пропорции числа детей у родителей);
1.2.5. Выбор наилучшего решения и запись его характеристик;
End while
1.3. Нормировка числа копий мемов в соответствии с размером ИП;
2. Определение типа мемов (группы 1—4), где к группе 1 относят мемы, для которых характерен рост числа копий мема в течение трех рассматриваемых периодов, к группам 2—3 — мемы, для которых характерны смены роста и спада, к группе 4 — мемы, для которых характерен спад в течение двух последних периодов;
2.1. If *если не группа 1* ***Then***
2.3. Выход1*: определяется вероятность, что будет хоть одна копия мема в следующий интервал времени (древовидная модель);*
2.3. Else
Прогноз логарифма числа копий (регрессионная модель);
2.4. Выход2*: определение интервала числа копий мема с 90 %-ной вероятностью (на основе распределения Su-Джонсона).*
End

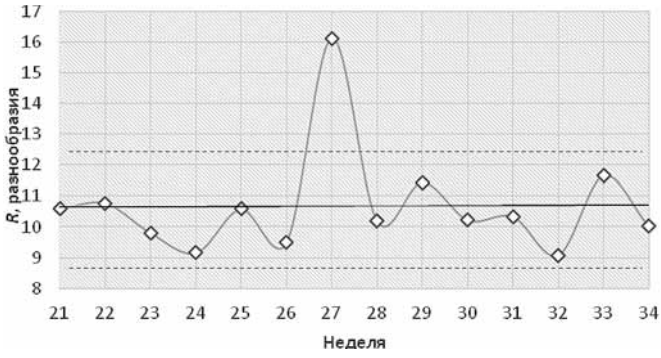


Рис. 3. Изменение коэффициента разнообразия *R* ИП: на 21—26-й, 28—34-й неделях в пределах нормы (эволюционный процесс), с 26-й по 28-ю недели — выше нормы (революционный процесс); штриховыми линиями показаны границы колебания коэффициента *R* (±1,5 от среднего значения около 11 ед.)

Оценка размера ИП — организационно и вычислительно трудоемкая задача, поэтому потребовалась эвристика, позволяющая избавиться от необходимости пересчитывать размер и состав ИП на каждом эволюционном этапе. С этой целью автором были определены два ключевых условия описания процесса функционирования ИП: а) изменение мощности мультимножества элементов, представляющего ИП, определяется некоторым известным законом; б) доля элементов-потомков в ИП пропорциональна доле элементов-родителей. Ввод новых условий потребовал оценки изменения качества прогнозов СИС с применением мем-грамм-модели. Указанные условия явились следствием изучения данных об изменении содержания реального ИП [17]. На рис. 3 представлен один из главных выводов, а именно — наличие границ изменения коэффициента разнообразия в ИП, определяемого как среднее число копий *n*-грамм, которое приходится на один мем.

В соответствии с принятыми положениями формализуются условия для оценки ошибки прогнозов мем-грамм-модели.

Пусть выполняются следующие условия.

A. Задана последовательность непустых конечных мультимножеств C_k, $k = 1, ..., n+1$, $n \in \mathbb{N}$ и $n > 1$, состоящих из произвольного числа элементов *w*, именуемых словами. Известен состав слов мультимножеств C_k для $\forall k \leqslant n$. Задано конечное множество слов W такое, что для всех $k = 1, ..., n+1$ $\forall w \in C_k$, $w \in W$.

B. Заданы отношения схожести между любыми двумя элементами из множества слов. Запись вида $\tilde{w} \sim w$ означает, что слово $\tilde{w}$ похоже на слово w в соответствии с заданными критериями схожести. Критерий схожести определяется целями конкретной задачи. Например, для *n*-граммных моделей это схожесть части элементов одной *n*-граммы с частью элементов другой *n*-граммы, а для векторных моделей — косинусная мера близости векторов слов относительно выбранного контекста.

C. Число элементов в мультимножестве C_k определяется следующим образом:

— для одинаковых элементов $w_i = w_i$, $w_i \in C_k$,

$$N^k_{w_i} = N^k_{\tilde{w}=w_i} = \sum_{w_i \in C_k} 1,$$

— для подобных элементов $\tilde{w}_i = \tilde{w} \sim w_i$, $\tilde{w} \in C_k$,

$$N^k_{\tilde{w}_i} = N^k_{\tilde{w}\sim w_i} = \sum_{\tilde{w}\sim w_i \in C_k} 1.$$

Легко заметить, что $1 \leqslant N^k_{w_i} \leqslant N^k_{\tilde{w}\sim w_i}$. При этом $N^k_{\tilde{w}\sim w_i} = N^k_{w_i}$ тогда и только тогда, когда множество подобных элементов состоит из одного элемента w_i. Будем называть такие слова самоподобными.

D. Задан функционал $\varphi : \{2, ..., n+1\} \to U$, определяемый следующим правилом:

$$\varphi(k) = \left\{ \tilde{M}_{k-1} \subseteq M_{k-1} \asymp \operatorname{supp}\left(C_{k-1} \cap C_k\right) \right\},$$

где $M_{k-1} = \left\{m_j\right\}_{j=1}^{|M_{k-1}|}$ — множество элементов наследственности для мультимножества C_k, U — универсум-множество, образованное из всех возможных (уникальных) вариантов комбинации элементов множества W.

E. Задан алгоритм λ прогноза числа $N^k_{w_i^m}$ элементов наследственности $w_i^m \in \tilde{M}_{k-1} \subseteq M_{k-1}$ при $k = 2, ..., n+1$. На вход алгоритма поступает множество элементов w_i^m, где для каждого слова определен ряд значений $N^{k-1}_{w_i}$ при $k = 2, ..., n+1$. На выходе алгоритма определяется число копий $N^k_{w_i^m}$ слова w_i^m в мультимножестве C_k. Отметим, что на практике оценка числа копий является интервальной.

F. Задано семейство функционалов P_k над множеством элементов w_i^m мультимножества C_k так, что $W \xrightarrow{P_k} [0; 1]$. Пусть $k = 2, ..., n+1$ и $w_i^m \in C_k$, тогда $P_k\left(w_i^m\right) = \dfrac{N^k_{w_i^m}}{N^k_{\tilde{w}_i}}$ — относительная частота элемента (мем-граммы) w_i^m. В соответствии с условием C выполняется двойное неравенство $0 < P_k\left(w_i^m\right) \leqslant 1$ для всех $w_i^m \in C_k$ и $\sum\limits_{\tilde{w}} P_k\left(w_i^m\right) = 1$ для всех $\tilde{w} \sim w_i^m \in C_k$.

G. Будем называть элементами изменчивости мультимножества C_k элементы $w_i^{\text{var}} \in C_k$, если данные слова отвечают также условию $w_i^{\text{var}} \notin M_{k-1}$. Это условие неявно определяет множество элементов изменчивости.

H. Для всех элементов изменчивости w_i^{var} выполняется тождество

$$\frac{N^k_{w_i^{\text{var}}\sim w_i^m}}{\sum\limits_{w_i^{\text{var}}} N^k_{w_i^{\text{var}}}} \equiv \frac{N^k_{w_i^m}}{\sum\limits_{\tilde{w}} N^k_{\tilde{w}\sim w_i^m}}.$$

Доля элементов изменчивости w_i^{var} тождественно равна доле наследуемых элементов w_i^m, подобных данным элементам. Это условие отражает правило — относительная частота наследников (выражение слева от знака равенства) пропорциональна относитель-

ной частоте родителей (выражение справа). Прогноз числа наследников определяется на основе данных прогноза размера популяции родителей, соответствующих критериям отбора. Данные параметры модели определяются в условиях E и D соответственно.

Постановка задачи

По известным мультимножествам C_1, C_2, ..., C_n оценить $P_{n+1}\left(w_i^m\right)$ и определить максимальную ошибку $\mathrm{Err}_{\max}\left[P_{n+1}\left(w_i^m\right)\right]$ данной оценки при условиях:

1) ограничения изменения мощности мультимножеств

$$\exists \alpha,\ n \in \mathbb{N},\ \varepsilon \in \mathbb{R},$$
$$0 < \varepsilon < \alpha,\ \forall k \in \{1,\ldots,n+1\} : |C_{n+1}| \in [\alpha - \varepsilon;\ \alpha + \varepsilon];$$

2) сохранения доли наследственности в мультимножествах

$$\exists \beta,\ \xi \in \mathbb{R}^+,\ n \in \mathbb{N},\ 0 < \xi < \beta,$$
$$\forall k \in \{1,\ldots,n+1\} : \frac{|C_{n+1}|}{|M_n|} \in [\beta - \xi;\ \beta + \xi].$$

Учесть, что в множестве C_{n+1} могут быть элементы изменчивости $w_i^{\mathrm{var}} \notin C_n$, общее число которых и количество экземпляров каждого из этих элементов неизвестно.

Далее в статье для краткости выражение $|C_{n+1}| \in [\alpha - \varepsilon;\ \alpha + \varepsilon]$ будем записывать в виде $|C_{n+1}| = \alpha \pm \varepsilon$, а выражение $\frac{|C_{n+1}|}{|M_n|} \in [\beta - \xi;\ \beta + \xi]$ — в виде $\frac{|C_{n+1}|}{|M_n|} = \beta \pm \xi$.

Решение

Топологическое представление задачи удобно продемонстрировать в виде пересекающихся попарно шаров, представляющих мультимножества различной мощности (рис. 4, см. третью сторону обложки). Такое представление аналогично диаграммам Эйлера—Венна для множеств. Объем шара характеризует мощность мультимножества C, общий объем пересекающихся шаров определяет нижнюю границу общего числа элементов двух мультимножеств, площадь поверхности (круга) пересечения определяет опорное множество M общих элементов мультимножеств. Решение задачи предполагает оценку ошибки определения минимального объема пространства, заключенного в пересечении шаров, для заданной площади поверхности круга, образующейся в сечении этого пересечения плоскостью, проходящей через линию (поверхности) пересечения данных шаров.

При условии 1. Из условий A, B, C, D, E следует, что $\tilde{M}_n \subseteq \mathrm{supp}(C_k)$, $\lambda\left(w_i^m\right) = N_{w_i^m}^k$, т. е. известен частичный состав элементов наследственности $\tilde{M}_n$, но неизвестно число элементов наследственности C_k и изменчивости w_i^{var}. Исходя из условия F семейство функционалов $P_k\left(w_i^m\right)$ для $w_i^m = m_j$ можно записать следующим образом:

$$P_k\left(w_i^m\right) = \frac{N_{w_i^m}^k}{N_{\tilde{w} \sim w_i^m}^k + N_{w_i^{\mathrm{var}} \sim w_i^m}^k}. \quad (1)$$

Отметим, что равенство (1) будет выполнено и в случае, если считать элементами изменчивости все общие подобные элементы множества M_n, которых нет в множестве $\tilde{M}_n$:

$$\tilde{w} \in C_k \mid \tilde{w} \notin \tilde{M}_n \to \tilde{w} \notin C_n \Leftrightarrow \tilde{w} \notin M_n.$$

Число элементов наследственности $N_{w_i^m}^k$ и подобных элементов $N_{\tilde{w} \sim w_i^m}^k$ известно по условию. Количество элементов изменчивости $N_{w_i^{\mathrm{var}} \sim w_i^m}^k$, подобных элементам наследственности, неизвестно. Исходя из условия F и $|C_k| = \sum\limits_{w_i \in \mathrm{supp}\{C_k\}} N_{w_i}^k = \sum\limits_{w_i^{\mathrm{var}}} N_{w_i^{\mathrm{var}}}^k + \sum\limits_{w_i^m} N_{w_i^m}^k$, следует

$$N_{w_i^{\mathrm{var}} \sim w_i^m}^k = \frac{N_{w_i^m}^k}{\sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k} \sum_{w_i^{\mathrm{var}}} N_{w_i^{\mathrm{var}}}^k =$$
$$= \frac{N_{w_i^m}^k}{\sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k} \left(|C_k| - \sum_{w_i^m} N_{w_i^m}^k \right).$$

Подстановка полученного выражения в выражение (1) приводит к выражению

$$P_k\left(w_i^m\right) = \frac{N_{w_i^m}^k}{N_{\tilde{w} \sim w_i^m}^k + \frac{N_{w_i^m}^k}{\sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k} \left(|C_k| - \sum\limits_{w_i^m} N_{w_i^m}^k \right)} = \frac{N_{w_i^m}^k \sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k}{|C_k|\, N_{\tilde{w} \sim w_i^m}^k}. \quad (2)$$

Учитывая условие задачи $|C_k| = \alpha \pm \varepsilon$, получим оценку $P_k\left(w_i^m\right)$:

$$\frac{N_{w_i^m}^k}{N_{\tilde{w} \sim w_i^m}^k} \frac{\sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k}{(\alpha + \varepsilon)} \leqslant P_k\left(w_i^m\right) \leqslant \frac{N_{w_i^m}^k}{N_{\tilde{w} \sim w_i^m}^k} \frac{\sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k}{(\alpha - \varepsilon)}. \quad (3)$$

Утверждение 1. Максимальная ошибка при оценке $P_{n+1}\left(w_i^m\right)$ зависит от мощности мультимножества $|C_{n+1}| = \alpha \pm \varepsilon$ и определяется следующим выражением:

$$\mathrm{Err}_{\max}\left[P_{n+1}\left(w_i^m\right)\right] \leqslant \frac{N_{w_i^m}^{n+1}}{N_{\tilde{w} \sim w_i^m}^{n+1}} \left(\frac{2\varepsilon \sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^{n+1}}{\alpha^2 - \varepsilon^2} \right). \quad (4)$$

Действительно, выражение (4) легко получается из (3) нахождением Евклидова расстояния между максимальными и минимальными значениями $P_k\left(w_i^m\right)$:

$$\mathrm{Err}_{\max}\left[P_k\left(w_i^m\right)\right] = \frac{N_{w_i^m}^k \sum\limits_{\tilde{w}} N_{\tilde{w} \sim w_i^m}^k}{N_{\tilde{w} \sim w_i^m}^k} \sqrt{\left(\frac{1}{\alpha - \varepsilon} - \frac{1}{\alpha + \varepsilon} \right)^2}.$$

Принимая во внимание справедливость условия $\alpha - \varepsilon > 0 \Rightarrow \alpha^2 - \varepsilon^2 > 0$, никаких дополнительных ограничений не требуется. Извлекая из-под корня исходное выражение, получим (4).

При условии 2. В данном случае ситуация осложняется тем фактором, что мощность мультимножества $|C_k|$ не постоянна, а изменяется пропорционально $|M_{k-1}|$. Поскольку

$$\tilde{M}_{k-1}\{m_1, m_2, \ldots, m_j\} \subseteq M_{k-1} \asymp \operatorname{supp}(C_{k-1} \cap C_k),$$

то условие D не дает гарантии того, что в полученном множестве $\tilde{M}_{k-1}$ есть все элементы множества M_{k-1}. Поэтому воспользоваться $\frac{|C_k|}{|M_{k-1}|} = \beta \pm \xi$ пока нельзя. Необходим "легальный" способ построения $\frac{|C_k|}{|M_{k-1}|} = \frac{|C'_k|}{|\tilde{M}_{k-1}|}$ при $\tilde{M}_{k-1} \subseteq M_{k-1}$, тогда задача сведется к решению задачи при условии 1.

Введем следующее определение.

Определение 1. Мультимножества с общими элементами C' и C будем называть подобными $\left(C' \underset{\sim}{K} C\right)$ с коэффициентом подобия $K \in \mathbb{R}$ относительно всех элементов $M = \operatorname{supp}(C' \cap C)$, если $\forall m \in M$ выполняется условие $N_{w=m} = KN_{w'=m}$, где N_w — значение функции-счетчика числа элементов w в мультимножестве, которая определена в условии C.

Построим доказательство необходимого следствия (сведение к условию 1) на серии утверждений.

Утверждение 2. Если $C' \underset{\sim}{K} C$ и $C, C', C'' \neq \varnothing$, $M = \operatorname{supp}(C'' \cap C) \neq \varnothing$, $M' = \operatorname{supp}(C' \cap C) \subseteq M$, то для выполнения $\frac{|C|}{|M|} = \frac{|C'|}{|M'|}$ необходимо и достаточно, чтобы существовало число $K = \frac{|M|}{|M'|} = \frac{|C|}{|C'|}$.

Доказательство.

Достаточность. Пусть $K = \frac{|M|}{|M'|} = \frac{|C|}{|C'|}$, тогда $\frac{|C'|}{|M'|} = \frac{|C|}{|M|}$.

Необходимость. Из условия $\frac{|C'|}{|M'|} = \frac{|C|}{|M|}$ следует существование $K = \frac{|M|}{|M'|} = \frac{|C|}{|C'|}$.

Утверждение 3. Если $C' \underset{\sim}{K} C$ и $C, C', C'' \neq \varnothing$, $M = \operatorname{supp}(C'' \cap C) \neq \varnothing$, $M' = \operatorname{supp}(C' \cap C) \subseteq M$, $K = \frac{|M|}{|M'|} = \frac{|C|}{|C'|}$, то $P(w') = P(w)$, где $w' \in C'$, $w \in C$.

Доказательство. $P(w') = \frac{N_{w'=w}}{N_{\tilde{w}'=w}}$, аналогично $P(w) = \frac{N_{w=w'}}{N_{\tilde{w}=w'}}$. В соответствии с определением подобия $C' \underset{\sim}{K} C$ следует $N_{w=w'} = KN_{w'=w}$, а также $N_{\tilde{w}=w} = KN_{\tilde{w}'=w}$, откуда получаем $P(w) = \frac{N_{w=w'}}{N_{\tilde{w}=w'}} = \frac{KN_{w'=w}}{KN_{\tilde{w}'=w}} = P(w')$.

Определение 2. Подмультимножество мультимножества, подобное этому мультимножеству, будем называть вложенным мультимножеством.

Теорема Расторгуева* о мощности вложенного мультимножества. Мощность вложенного мультимножества пропорциональна числу общих элементов с мультимножеством, в которое оно вложено, и обратно пропорциональна коэффициенту подобия данных мультимножеств.

Доказательство.

Пусть C', C — подобные мультимножества относительно M', $M' = \operatorname{supp}(C' \cap C) \neq \varnothing$ и $C' \subseteq C$. В соответствии с определением подобных мультимножеств для каждого $m \in M'$ выполняется $\sum_{m=1}^{|M'|} N_{w=m} = K \sum_{m=1}^{|M'|} N_{w'=m}$.

Из $C' \subseteq C \Rightarrow \operatorname{supp} C' = M' \Rightarrow \sum_{m=1}^{|M'|} N_{w'=m} = |C'|$, откуда окончательно получаем $|C'| = \frac{\sum_{m=1}^{|M'|} N_{w=m}}{K}$.

Следствие. Применяя теорему 1 и учитывая утверждение 2, получим, что для выполнения условия $\frac{|C'_k|}{|\tilde{M}_{k-1}|} = \frac{|C_k|}{|M_{k-1}|}$ (равнозначного $\beta - \xi \leqslant \frac{|C'_k|}{|\tilde{M}_{k-1}|} \leqslant \frac{|C_k|}{|M_{k-1}|}$) при построении подобного множества $C'_k \subseteq C_k$ относительно $\tilde{M}_{k-1} \subseteq M_{k-1}$ достаточно задать коэффициент подобия $K = \frac{\sum_{i=1}^{|\tilde{M}_{k-1}|} N^k_{w_i^m}}{|\tilde{M}_{k-1}| \operatorname{const}}$, где $w_i^m \in \tilde{M}_{k-1} \subseteq M_{k-1}$.

Данное выражение выводится при решении следующей системы уравнений:

$$\begin{cases} |C'_k| = \dfrac{\sum_{m=1}^{|M'|} N_{w=m}}{\operatorname{const}} \\ \dfrac{|C_k|}{|C'_k|} = \operatorname{const} \\ \dfrac{|M_{k-1}|}{|\tilde{M}_{k-1}|} = \operatorname{const} \\ |C_k| = \operatorname{const} |M_{k-1}|. \end{cases}$$

* Теорема названа в память о д-ре техн. наук, проф. Сергее Павловиче Расторгуеве, который является автором идеи о вложенности информационных пространств [18].

Утверждение 4. Максимальная ошибка при оценке $P_{n+1}\left(w_i^m\right)$ обратно пропорциональна мощности множества $\tilde{M}_n$ и определяется следующим выражением:

$$\mathrm{Err}_{\max}\left[P_{n+1}\left(w_i^m\right)\right] \leqslant \frac{N_{w_i^m}^{n+1}}{N_{\tilde{w}\sim w_i^m}^{n+1}}\left(\frac{2\xi\sum\limits_{\tilde{w}} N_{\tilde{w}\sim w_i^m}^{n+1}}{\left|\tilde{M}_n\right|\left(\beta^2-\xi^2\right)}\right). \qquad (5)$$

Доказательство.

Построим мультимножество C'_{n+1}, подобное мультимножеству C_{n+1} относительно заданного $\tilde{M}_n$ так, чтобы выполнялось условие $P_{n+1}\left(w_i^m\right)=P_{n+1}\left(w_i^{\tilde{m}}\right)$. Для выполнения данного условия воспользуемся следствием теоремы Расторгуева и определим коэффициент подобия $K=\dfrac{\sum\limits_{i=1}^{|\tilde{M}_n|} N_{w_i^m}^{n+1}}{\left|\tilde{M}_n\right|\mathrm{const}}$, где const = β ± ξ. Учитывая выражение (2), запишем неравенство для $P_k\left(w_i^m\right)=P_k\left(w_i^{\tilde{m}}\right)$ при $k = n + 1$:

$$\frac{N_{w_i^m}^k}{N_{\tilde{w}\sim w_i^m}^k}\frac{\sum\limits_{\tilde{w}} N_{\tilde{w}\sim w_i^m}^k}{\left|\tilde{M}_n\right|(\beta+\xi)} \leqslant P_k\left(w_i^m\right) \leqslant \frac{N_{w_i^m}^k}{N_{\tilde{w}\sim w_i^m}^k}\frac{\sum\limits_{\tilde{w}} N_{\tilde{w}\sim w_i^m}^k}{\left|\tilde{M}_n\right|(\beta-\xi)},$$

откуда

$$\mathrm{Err}\left[P_k\left(w_i^m\right)\right] \leqslant \frac{N_{w_i^m}^k}{N_{\tilde{w}\sim w_i^m}^k}\left(\frac{2\xi\sum\limits_{\tilde{w}} N_{\tilde{w}\sim w_i^m}^k}{\left|\tilde{M}_n\right|\left(\beta^2-\xi^2\right)}\right).$$

Ввиду условия $\beta-\xi>0\Rightarrow\beta^2-\xi^2>0$, никаких дополнительных ограничений не требуется. Полученное выражение соответствует выражению (5) Утверждения 4.

Заключение

Задача оценки ошибки при прогнозе относительной частоты элементов мультимножества, которые содержатся в последовательности мультимножеств, была успешно решена при следующих трех условиях.

1. На основе истории изменения состава последовательности мультимножеств задан способ прогноза состава и числа элементов мультимножества, которые содержатся в следующем ближайшем мультимножестве.

2. Изменение мощности последовательных мультимножеств определяется одним из вариантов: а) в некотором фиксированном диапазоне; б) пропорционально мощности опорного множества мультимножества, образованного пересечением двух соседних мультимножеств.

3. Доля элементов (правого) мультимножества, подобных общим элементам двух мультимножеств (левого и правого), пропорциональна доле общих элементов в (правом) мультимножестве.

Таким образом, если признаковое пространство объектов для машинного обучения может быть представлено в виде изменяющихся мультимножеств и характер этих изменений может быть задан эволюционным алгоритмом, то при определении модели прогноза изменений числа копий некоторой части общих элементов мультимножеств можно оценить ошибку прогноза относительной частоты данных элементов, даже не зная всего состава мультимножества.

Подобный подход открывает новую перспективу при решении задачи оценки изменения качества прогнозов модели, связанного со смещением верифицирующих данных — концептуальным смещением. Главным преимуществом предложенного подхода перед альтернативными методами является относительная простота его реализации на практике. Такой подход позволяет не переучивать модель при каждом обнаружении смещения данных, а дает возможность прогнозировать и учитывать ошибку, с ним связанную. Необходимость переобучения модели возникает только тогда, когда ошибка прогнозов неприемлема.

Автор выражает благодарность д-ру физ.-мат. наук, проф. МГУ имени М. В. Ломоносова Валерию Александровичу Васенину, д-ру техн. наук, проф. Сергею Павловичу Расторгуеву и канд. физ.-мат. наук, ст. науч. сотр. механико-математического факультета МГУ имени М. В. Ломоносова Алексею Владимировичу Галатенко за помощь в разработке мем-грамм-модели.

Список литературы

1. **Воронцов К. В.** Математические методы обучения по прецедентам (теория обучения машин). URL: http://www.machinelearning.ru/wiki/images/6/6d/voron-ml-1.pdf

2. **Гудфеллоу Я., Иошуа Б., Курвилль А.** Глубокое обучение. М.: Издательство ДМК-Пресс, 2017. 652 с.

3. **Босс В.** Лекции по математике. Т. 4. Вероятность, информация, статистика. Изд. 2-е. М.: ЛКИ, 2008. 210 с.

4. **Moreno-Torres J. G., Llorà X., Goldberg D. E., Bhargava R.** Repairing Fractures between Data using Genetic Programming-based Feature Extraction: A Case Study in Cancer Diagnosis // Information Sciences. 2013. Vol. 222. P. 805—823. DOI: 10.1016/j.ins.2010.09.018.

5. **Shubham Jain.** Covariate Shift — Unearthing hidden problems in Real World Data Science. Analytics Vidhya. 10.07.2017. URL: https://www.analyticsvidhya.com/blog/2017/07/covariate-shift-the-hidden-problem-of-real-world-data-science/

6. **Wang K., Zhou S., Fu C. A., Yu J. X., Jerey F., Yu X.** Mining changes of classification by correspondence tracing by correspondence tracing // Proceedings of the 2003 SIAM International Conference on Data Mining. 2003. P. 95—106.

7. **Yang Y., Wu X., Zhu X.** Conceptual equivalence for contrast mining in classification learning // Data & Knowledge Engineering. 2008. Vol. 67, No. 3. P. 413—429.

8. **Cieslak D. A., Chawla N. V.** A framework for monitoring classiffiers' performance: when and why failure occurs? and why failure occurs? // Knowledge and Information Systems. 2009. Vol. 18, No. 1. P. 83—109.

9. **Moreno-Torres J. G., Raeder T., Alaiz-RodríGuez R., Chawla N. V., Herrera F.** A unifying view on dataset shift in classification // Pattern Recognition. 2012. Vol. 45, No. 1. P. 521—530.

10. **Brown P. F., Dellapietra V., de Souza P. V.** et al. Class-based n-gram models of natural language // Computational linguistics 1992. Vol. 18, No. 4. P. 467—479.
11. **Leskovec J., Backstrom L., Kleinberg J.** Meme-tracking and the dynamics of the news cycle // Proceedings of the 15th ACM SIGKDD — International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2009. P. 497—506.
12. **Mikolov T., Chen K., Corrado G., Dean J.** Efficient estimation of world representation in vector space. 2013. URL: http://arxiv.org/pdf/1301.3781.pdf
13. **Артемов А. А.** Модель оценки уровня угроз информационных вызовов плану содержания информационного пространства социально-телекоммуникационной системы // Информационные войны. 2015. № 3. С. 83—97.
14. **Петровский А. Б.** Многокритериальное принятие решений по противоречивым данным: подход теории мультимножеств // Информационные технологии и вычислительные системы. 2004. № 2. С. 56—66.
15. **Петровский А. Б.** Пространства множеств и мультимножеств. М.: Едиториал УРСС, 2003. 248 с.
16. **Артемов А. А.** Эксперимент по моделированию процесса эволюции содержания информационного пространства социума (с применением мем-грамм-модели) // Программная инженерия. 2016. Т. 7, № 7. С. 291—306.
17. **Артемов А. А., Галатенко А. В.** Моделирование процесса эволюции содержания информационного пространства социума (мем-грамм-модель) // Программная инженерия. 2017. Т. 8, № 1. С. 511—523.
18. **Расторгуев С. П.** Практические аспекты теории вложенности пространств. Центр стратегических оценок и прогнозов. М.: АНО ЦСОиП, 2017. 92 с.

A Predicative Estimation of Supremum of the Model's Forecast Error Resulting from a Conceptual Dataset Shift. On the Example of the Meme-gram-model

A. A. Artemov, artemsince2@ya.ru, Center of Strategic Nuclear Forces Research at Academy of Military Sciences, Moscow Region, Jubileiny, 141090, Russian Federation

Corresponding author:

Artemov Artem A., Research associate, Center of Strategic Nuclear Forces Research at Academy of Military Sciences, Moscow Region, Jubileiny, 141090, Russian Federation,
E-mail: artemsince2@ya.ru

Received on February 26, 2018
Accepted on April 09, 2018

The paper presents a solution to the problem of estimating the forecast error of statistical probability of presence of a feature sequence in the objects verifying the model of the studied object domain. The author considers object's feature sequence as n-grams. The object is an n-gram of fixed length. A population of "mutating" objects, united by a given criterion (e.g. time), forms evolving multisets of variative power. With such formalization, a common element of two multisets or meme-gram identifies the presence of a feature sequence. The probability of such event is defined as a functional of the number of copies of a meme-gram. The model in which this axiomatics is determined is called a meme-gram model.

The presented solution focuses on the issue of estimating relative frequency of repeating elements of multisets with the condition of possibility of forecasting their number only for a part of these elements. The proposed solution is particularly in demand in the field of creating knowledge representation models for self-learning systems in the condition of limited amount of training examples from the total volume of permanently changing Big Data.

Keywords: *Meme-gram model, m-gram, data shift, elements of heredity, memetic algorithm, evolving system of knowledge, similar multisets, embedded multisets, Rastorguev's theorem*

For citation:

Artemov A. A. A Predicative Estimation of Supremum of the Model's Forecast Error Resulting from a Conceptual Dataset Shift. On the Example of the Meme-gram-model, *Programmnaya Ingeneria,* 2018, vol. 9, no. 6, pp. 272—280.

DOI: 10.17587/prin.9.272-280

References

1. **Vorontsov K. V.** *Matematicheskie metody obucheniya po pretsedentam (teoriya obucheniya mashin)* (Mathematical methods of learning by precedents (theory of machine learning)), available at: http://www.machinelearning.ru/wiki/images/6/6d/voron-ml-1.pdf (in Russian).
2. **Gudfellou Ya., Ioshua B., Kurvill A**. Glubokoe obuchenie (Deep learning), Moscow, Litres, 2017, 652 p. (in Russian).
3. **Boss V.** *Lektsii po matematike* (Lectures on mathematics), T. 4. Veroyatnost', informatsiya, statistika. Izd. 2, Moscow, LKI, 2008, 210 p. (in Russian).
4. **Moreno-Torres J. G., Llorà X., Goldberg D. E., Bhargava R.** Repairing Fractures between Data using Genetic Programming-based Feature Extraction: A Case Study in Cancer Diagnosis., *Information Sciences,* 2013, vol. 222, pp. 805—823, DOI: 10.1016/j.ins.2010.09.0185.

5. **Shubham Jain.** Covariate Shift — Unearthing hidden problems in Real World Data Science. Analytics Vidhya. 10.07.2017, available at: https://www.analyticsvidhya.com/blog/2017/07/covariate-shift-the-hidden-problem-of-real-world-data-science/

6. **Wang K., Zhou S., Fu C. A., Yu J. X., Jerey F., Yu X.** Mining changes of classification by correspondence tracing by correspondence tracing, *Proceedings of the 2003 SIAM International Conference on Data Mining*, 2003, pp. 95—106.

7. **Yang Y., Wu X., Zhu X.** Conceptual equivalence for contrast mining in classification learning, *Data & Knowledge Engineering*, 2008, vol. 67, no. 3, pp. 413—429.

8. **Cieslak D. A., Chawla N. V.** A framework for monitoring classiffiers' performance: when and why failure occurs? and why failure occurs? *Knowledge and Information Systems*, 2009, vol. 18, no. 1, pp. 83—109.

9. **Moreno-Torres J. G., Raeder T., Alaiz-RodríGuez R., Chawla N. V., Herrera F.** A unifying view on dataset shift in classification, *Pattern Recognition*, 2012, vol. 45, no. 1, pp. 521—530.

10. **Brown P. F., BroDellapietra V., de Souza P. V., Lai J., Mercer R.** Class-based n-gram models of natural language, *Computational linguistics*, 1992, vol. 18, no. 4, pp. 467—479.

11. **Leskovec J., Backstrom L., Kleinberg J.** Meme-tracking and the dynamics of the news cycle, *Proceedings of the 15th ACM SIGKDD — International Conference on Knowledge Discovery and Data Mining*, New York, ACM, 2009. P. 497—506.

12. **Mikolov T., Chen K., Corrado G., Dean J.** Efficient estimation of world representation in vector space, available at: http://arxiv.org/pdf/1301.3781.pdf

13. **Artemov A. A.** Model' ocenki urovnja ugroz informacionnyh vyzovov planu soderzhanija informacionnogo prostranstva social'no-telekommunikacionnoj sistemy (A Model for Assessing The Level of Threat of Information Challenges To The Content Plan of The Socio-Telecommunication System Information Space), *Informacioniy vojni,* 2015, no. 3, pp. 83—97. (in Russian).

14. **Petrovskiy A. B**. Mnogokriterial'noe prinyatie resheniy po protivorechivym dannym: podkhod teorii mul'timnozhestv (Multicriteria decision-making on contradictory data: the approach of the theory of multisets), *Informatsionnye tekhnologii i vychislitel'nye sistemy*, 2004, no. 2, pp. 56—66 (in Russian).

15. **Petrovskiy A. B.** P*rostranstva mnozhestv i mul'timnozhestv (Spaces of sets and multisets).* Moscow, Editorial URSS, 2003, 248 p. (in Russian).

16. **Artemov A. A.** Jeksperiment po modelirovaniju processa jevoljucii soderzhanija informacionnogo prostranstva sociuma (s primeneniem mem-gramm-modeli) (An Experiment on Modeling Society Information Space Evolution Process (by using Meme-Gram-Model)), *Programmnaya Ingeneria,* 2016, vol. 7, no. 7, pp. 291—306 (in Russian).

17. **Artemov A. A., Galatenko A. V.** Modelirovanie protsessa evolyutsii soderzhaniya informatsionnogo prostranstva sotsium*a* (Mem-gramm-model') (Modeling the process of evolution of the content of the information space of society (Mem-gram model)), *Programmnaya Ingeneria,* 2017, vol. 8, no. 11, pp. 511—523 (in Russian).

18. **Rastorguev S. P.** *Prakticheskie aspekty teorii vlozhennosti prostranstv* (Practical aspects of the theory of nesting spaces), Tsentr strategicheskikh otsenok i prognozov, Moscow, ANO TsSOiP, 2017, 92 p. (in Russian).

ИНФОРМАЦИЯ

Программный комитет конференции **SECR "Разработка ПО" 2018**,
которая состоится в Москве
12—14 октября 2018 г., приглашает докладчиков

Процедура приема заявок на доклады, мастер-классы, научные статьи (статьи принимаются только с докладом).

1. Подаете заявку не позднее 16 июля 2018.
2. Обсуждаете детали с куратором из программного комитета, дополняете заявку.
3. Программный комитет выбирает заявки путем голосования. Вас могут попросить доработать заявку, статью, пройти прослушивание для вынесения окончательного решения.
4. Готовите с куратором доклад. Прослушивание доклада куратором настоятельно рекомендовано для тех, кто выступает впервые или 1—2 раза в год.

Время на доклад: от 15 или 30 минут, на мастер-класс: от 1 до 8 часов.
Язык выступления: русский или английский. Слайды и статьи могут быть как на русском, так и на английском. Для статей английский язык предпочтителен.

Подробности на сайте конференции https://2018.secrus.org/

DOI:10.17587/prin.9.281-286

A. Yu. Popov, Associate Professor, alexpopov@bmstu.ru, Bauman State Technical University, Moscow, 105005, Russian Federation, **S. A. Belov,** Acting University Relations Manager, Sergey_Belov@ru.ibm.com, **A. V. Sorokin,** University Relations Manager in Russia and CIS countries, alexander_sorokin@ru.ibm.com, IBM Corporation, IBM East Europe/Asia Ltd., Moscow, 123112, Russian Federation

Corresponding author:

Popov Aleksey Yu., Associate Professor, Bauman State Technical University, Moscow, 105005, Russian Federation, E-mail: alexpopov@bmstu.ru,

Received on April 03, 2018
Accepted on April 12, 2018

Cloud-Based IT Learning Infrastructure to Support New Generation of Services

As other technical universities, Bauman University is challenged with timely incorporating newly emerged information technologies, such as Internet of Things (IoT), in education and research. One of those challenges for universities today is implementation of cloud technologies to build flexible infrastructure to support education and research in wide area of disciplines.

Since the cloud has become the primary platform to host IoT services, universities need to integrate cloud-based resources into the IT infrastructure to support student projects. The target cloud platform has to be IoT enabled: it should be scalable to be connected with many thing instances; flexible to implement new features and services; and simple to reduce the development time.

In this paper we present an experience and example of implementation of multi-vendor cloud infrastructure to support modern research activities. Various cloud technologies implemented to improve educational process will be considered and some future work will be discussed on the example of Bauman University.

Keywords: *cloud technology, IBM system, educational process, Bauman university, IBM Cloud cloud, cloud computing, learning management system, Internet of Things, systems of systems*

Introduction

Within the fast changing IT, technical universities are facing a problem of timely incorporating of the newly emerged technologies in educational process. It requires deep revision of the current approach used in technical education, particularly in IT. One of the basic challenges for universities today is implementation of cloud technologies to build flexible education infrastructure.

It goes without saying that new information technologies like Internet of Things open new opportunities for cross disciplinary research both fundamental and applied. Particularly Bauman University is regularly approached by representatives from high-tech industry with proposals for joint research of IoT applications in different areas from telecom to agriculture. IoT are also very much attractive for new generation of students who actively discover new business opportunities supported by IoT.

It is clear that new opportunities arising due to tremendous quantity of different devises connected together [1] will be drastically transforming current IT marketplace. At the same time relatively high implementation cost, IP protection, and great spectrum of constantly changing technologies make up-to-date research and education at technical universities quite challenging. It inevitably leads to necessity of close collaboration between industry and academia.

Among noticeable Bauman university achievements in this area was implementation of public cloud solutions with convenient infrastructure and helpful analytics to support IoT projects. IoT laboratory established in Bauman University supported several noticeable IoT projects also by providing effective cloud infrastructure and necessary education materials for a quick start.

In particular, the new course devoted to IBM Cloud for IoT has been developed in 2014 and already became popular among students and professors. Several IoT projects

including international projects in collaboration with European universities in the areas of telecom, agriculture and last but not least smarter planet/smarter cities are currently being developed in the lab. In the coming talk we plan to discuss results achieved so far.

Project objectives

Device Democracy concept [2] implies that the information technologies market will be dramatically changing during the next two decades due to the huge amount of computing devices to be working coherently in core applications for a broad spectrum of–tasks in industries from mobile and consumer electronics to transportation and medicine. This trend is directly related to the University's future as well. Higher schools have to implement modern technologies in education to be competitive in modern education market. This is a great challenge for any technical school to solve. For example, some future technologies, such as IoT, may be too expensive for university to implement on demand. Development of prototypes, as well as laboratory infrastructure itself can be expensive. At the same time, new information technologies emerge rapidly and permanently in modern world, usually having life cycle of just several years, which leads to a great challenge of having up-to-date education process in any technical university. Another essential aspect here is that fixed departments structure in technical university creates ownership problems, which in turn delay implementation of disruptive technologies in education.

To be competitive in global education market Bauman University recently undertook some innovative actions to address the mentioned above challenges. One possible answer found was the IBM Academic Centre of Excellence in cloud computing to bring together key IT key players.

One of the main objectives of the project is creation of cloud services to be used in the educational process and research. The results of this work may be used in other technical universities in Russia.

Latest trends in IT and their impact on educational process

The vision of future technology development presented here is based on IT evolution driving by businesses demands. Any IT produces data. And data are fuel for business processes. Hence, the construction of simple IT ontology will contend classes that provide interaction among three main IT entities: human, machine and system. We obtain a matrix that represents examples of nine classes (fig. 1).

Technological level of technology examples, that belong to each class, evolves from simple to sophisticated — from the upper left corner to the lower right one. In addition, each higher class may absorb and use all technology solutions of the lower classes. We fix three types of IT environment in which interaction is taking place: user-centered, machine-dominated and system integrated.

Technology that supports interaction between human and user-centered environment creates the *initial class.* It contains the social technology tools — e-mail, e-messengers, Skype, forums and so forth.

Second class presents the means for communications among machines and user-centered environment. It is built on language processing technique. It includes a broad spectrum of instruments — from various programming languages to NLP.

Users interact with systems through interfaces and examples of *third class* technologies include standard API, speech-gesture interfaces, Google glasses and so on.

The next — *fourth class* covers the technologies that help users to extract knowledge from huge amount of data generated by machine-dominated environment. This type

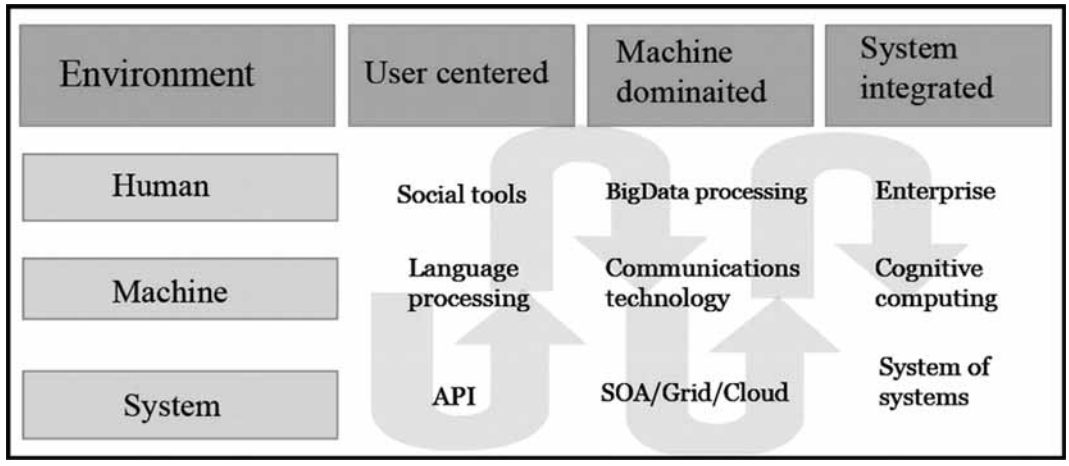


Fig. 1. Basic IT classes supporting interaction and data transfer

of technology embraces BigData processing instruments, searching engines, Q&A systems.

The *fifth class* helps the machines in interaction with machine-dominated domains. These are the communications technologies. The contemporary examples for them are: intelligent protocol and SDN.

The *sixth class* creates means that make possible for machine-dominated environment to support systems. The modern examples are grid and clouds based on SOA.

Enterprise is the *seventh class* of IT. ERP is an example. This class aggregates tools from all previous classes and supports users to implement their business processes in integrated systems.

Modern and especially future machines can also contribute in systems integration. Those that make it, are creating the eighth class. That kind of technology applies the AI and cognitive computing. This class includes the variety of smart devices, IoE, artificial intelligence mechanisms embedded in robots and drones (UAV), industrial and military robots for their navigation, control and mission execution. The latest IBM SyNAPSE chip which design was inspired by the human brain has ability to provide not just integrated systems with high-performance and low energy consumption. It will help to create new cognitive enterprises where machines will substitute human at least in routine operations in production, management and trade.

The last and highest class is the ninth: System of systems technologies. The most spectacular example of this type is Web itself. In future, we may expect appearing of highly integrated cross industrial fragments of Web with virtualized and variable architecture. The modern prototype of it is the Smart City as a system of systems based on IBM IOC.

Enterprise and corresponding technologies are the popular subjects of the courses in Technical University. The graduate students also need experience in solving the real life tasks and working on the big system platforms like IBM IOC.

BMSTU and IBM is a good example of the cooperation with IT industry: IBM Academic Center of Excellence is working since 2007, providing education of different IBM technologies from System z and zOS to WebSphere and DB2. Now it extends educational activity to Smart technologies, Clouds and IBM OpenPower initiative. Using cloud solutions IBM Academic Center of Excellence recently started several projects in areas of SmartCity and IoT. The IBM Intelligent Operations Center deployed at Bauman University purposed for education is also regarded as an important part of "education in project" concept.

Implementation of cloud technologies at Bauman University is going in several directions. Like a usual business organization, the university must manage its datacenter infrastructure, providing limited resources to a large number of interested faculties, students, as well as some other users. BMSTU datacenter is based on the IBM System z10 BC mainframe and tape library TS3500 [3]. There are also IBM HPC cluster, some System p and x86 servers. This "classical" enterprise datacenter (fig. 2) infrastructure allows BMSTU to use cloud solutions for several important subsystems:

- Instances of Learning Management Systems for educational content;

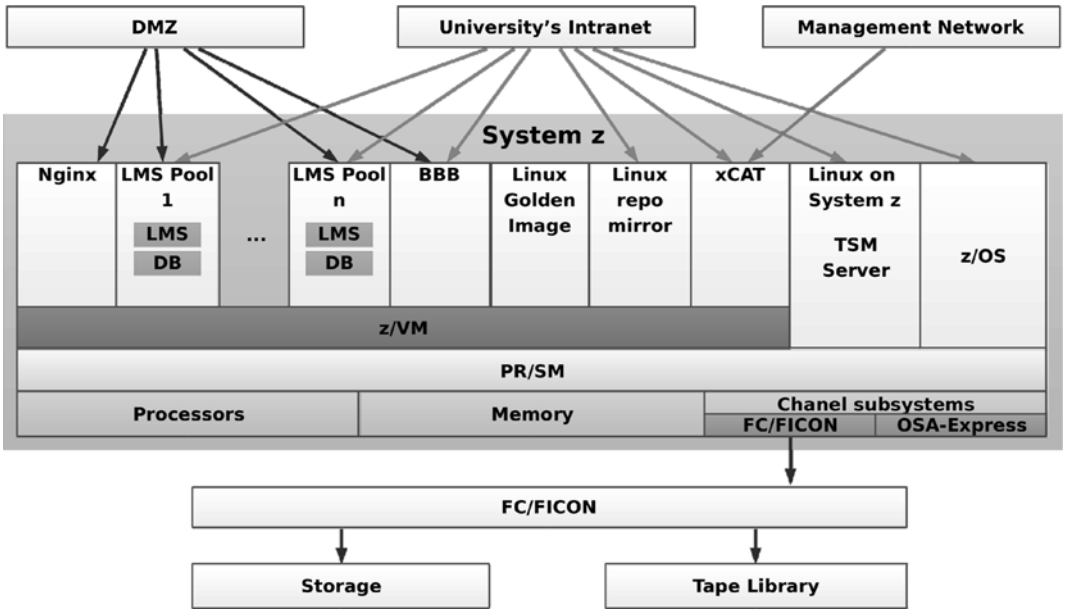


Fig. 2. Bauman University virtualized datacenter

- Cloud storage solution as a project repository;
- Version control system for software development;
- SmartCity IOC solution;
- Cloud infrastructure for IoT projects.

On infrastructure management level, BMSTU uses xCAT2 on IBM System z10 and VMware Vsphere and Xen on x86 servers. Such decisions allow providing efficiently services and resources for faculties.

BMSTU has deployed some important software products, based on the user access and cloud-based approaches. These solutions are used for student projects and software developments, and they are widely used in education process. This type of Cloud implementation (SAAS) of these products is useful [4], but requires sharing large pieces of resources, and maintenance of such products requires the involvement of highly qualified administrators. As examples we mention the "Smart City" situational centers solution: IBM Intelligent Operations Center [5]. Using xCAT2 cloud, Bauman University also deployed 89 instances of learning management system LMS Moodle and WSO2 cloud solution.

IoT laboratory, established in university, supports several noticeable IoT projects by providing effective cloud infrastructure (fig. 3) and necessary education materials for a quick start. Important BMSTU's achievements in this area are the implementation of public cloud with convenient infrastructure and vendor-supported PAAS for the IoT projects [6]. In these projects we have to support every developer individually, depending on what type of resources he needs: web hosting, computation resources, memory, storage, maintenance, documentation, best practice. For example, in IoT project, a developer is transferring telemetry information into cloud by any protocol to be stored in database, someone else is analyzing this data stream, visualizing results and then sending the control commands to remote equipment. This operational sequence requires a substantial amount of efforts to make it technically possible that leads to application of such technologies as VMWare Vsphere or Xen.

After considering the different cloud use cases we came to conclusion, that the hybrid cloud looks as a one of the best options for it: it allows the University to maintain in-house infrastructure and platforms and simultaneously with minimal effort and time allows communicating with public cloud domains wherever necessary. At the same time, the public cloud like IBM Cloud makes it possible to use many services for applications developed in university which is a great help for faculties and students involved in projects for IoT, mobile, data mining, web and others.

The new course devoted to IBM Cloud for IoT has been developed in 2014 and since that time stays popular among students and professors. Several IoT projects, including international projects in collaboration with European universities in the areas of telecom, agriculture and last but not least Smarter planet/Smarter cities are currently in progress.

It is understood that new information technologies like Internet of Things open new opportunities for cross disciplinary research, both fundamental and applied. The representatives from Russian high-tech industries regularly address the Bauman University with proposals for joint research in IoT area. IoT are also very much attractive for the new generation of students who actively discover new business opportunities supported by IoT.

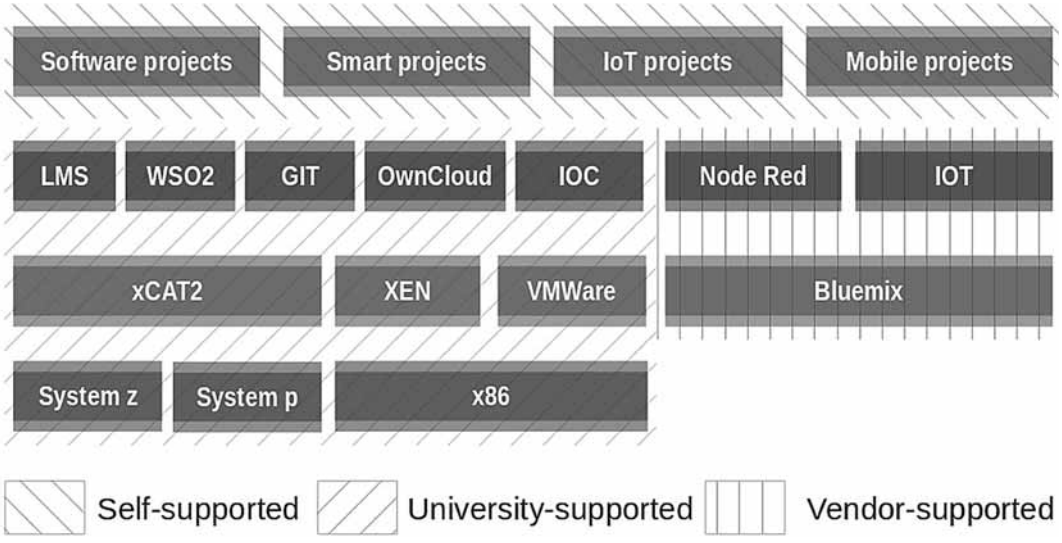


Fig. 3. Applications of Cloud technology at Bauman State Technical University

Project Examples

IoT laboratory. This project is aimed not only to create applicable set of equipment for IoT projects, but also to bring simple use cases of all parts of standards IoT solutions. The project uses the IBM Cloud services for data management and hardware platform RaspberryPi for telemetry data mining. The laboratory equipment includes oscilloscopes, generators, hardware computing platforms, sensors and network infrastructure. To support this project, we created a virtual community on LMS platform, which is also deployed on university cloud. The project includes several instances of used platforms, telemetry data collection, and examples of data processing in IBM Cloud from mobile platforms.

Poseidon project. BMSTU's students took part in the global Poseidon community, which is aimed at creating the technical solutions, educational practice and globally distributed infrastructure for water resources saving. The Poseidon project is a voluntary initiative, supported by the Dutch Courage Foundation, TU Delft and IBM Netherlands team, which aims to reduce water usage in the world. Due to this initiative, community developed educational material for monitoring the water in the soil of houseplants. The instrumental set for Poseidon applies IoT ideas, cloud computing, mobile, and analytics [7].

SmartCity project. BMSTU is working to understand the opportunities of introducing the technologies of the "Smart" class into the municipal processes: transportation, services and others activities. In this project BMSTU deployed IBM Intelligent Operations Center (IOC), which allows integrating SmartCity solution within the scalable platform. The prototype for this use case was created. The application for Android mobile platform transfers the information about monitored events by the MQTT Protocol to the IOC. The IOC operator receives all necessary information and can respond in accordance with KPI analysis. The obtained results can be used for control of multiple devices within the IoT concept. SmartCity community for students, academicians, IT professionals is a good example of the complex cloud technology application in education. In the period of learning in dynamic infrastructure and IBM IOC students are also developing new "Smart City" services. The virtual community, engaged in studying, supporting and developing of the IBM IOC services, uses xCAT2 cloud on System z and instance of LMS Moodle.

Smart crossroad. The crossroad control system analyses data from multiple sources to focus appropriate attention on citizen movement. At first, it uses the cars traffic from the GIS portal (maps.yandex.ru) and people traffic from city events system (afisha.yandex.ru). Data obtained from IBM Cloud help to predict people and cars traffic near the crossroad. Next, the system is sensing the around area to understand pedestrians allocation and light intensity. All this information used by Smart crossroad helps to show how dangerous the crossroad is at the moment. As a result, system controls smart backlights to attract attention of pedestrians and car drivers to make city crossroads more safe. In addition, video camera, attached to the system, captures vehicles near the stop line.

NRget(x). This system allows distributing energy flows from many sources to save energy for smart homes and smart cities. For that purpose, it analyses the data from external sources and the information from solar batteries, wind generators, accumulators to control the stored energy. System defines:

- How much energy is produced from power generators?
- What is the required power at the present moment?
- How much energy was stored before?

This allows defining how much energy should be stored, sold or bought for effective energy consumption.

Smart Pill Box. This project is for those people who have difficulties while taking medicine. The problem is that elderly people often miss the appointed time because of poor memory. People with disabilities such as bad eyesight sometimes mix pills up. The students team created a smart pill box which announces users when they should take their medicine and lights up the required compartment. The hardware consists of the pill box with sensors, LEDs, and buttons, while the software is based on the Cloud services and an Android application.

RealClimate. This project is dedicated to capture and replicate various climate conditions. The idea is to create the climate conditions in real time and equal to that one in another place. The project consists of three different modules. The first module is meant to be placed in natural conditions, collect data about the surrounding climate (temperature, humidity, soil moisture, etc.), and upload the collected data on the IBM Cloud. The second module is placed in the area, where the desired climate is to be replicated. It collects information about the second climate and uploads it on the Cloud. The third module is connected to various systems for recreating climate conditions (AC, irrigation, humidifier, etc.).

Information in the cloud from the two climate zones is compared, and commands are sent to the third module to enable/disable the necessary subsystems. This project has many potential use cases, four of which are listed below.

Zoos and arboretums: with the help of this product, stress associated with transporting plants and animals can be reduced to a minimum. The opposite effect can also be used for creating natural stress, which strengthens the immune system and increases vitality. This can be achieved by monitoring and recreating climate conditions in real-time.

- Industrial use: in farms and greenhouses for increasing crop yield.

- Virtual reality systems: for creating a more immersive experience.
- Science: for tracking and analyzing climate changes on planet Earth (and other planets in the future). This can also help forecast climate anomalies.

Results and future work

On the basis of experience of the cloud technologies implementation we come to the following conclusions:

- Cloud technologies make it possible to quickly and cost-effectively manage University IT infrastructure.
- University obtains deep knowledge and skills running special projects in hybrid cloud, which combines the private cloud deployed at the university's data center and external public clouds.
- IBM Cloud allows to use and integrate the disruptive technologies for data mining, IoT, mobile, analytic both in education and in research projects.
- The academic community needs to coordinate their activities and efforts with vendors. As a result of open discussions the most useful areas of research should be defined.

Further work with this project is aimed at the deep implementation of design-and-training method with cloud technologies. It needs large-scale deployment of cloud computing infrastructure, IoT laboratory, and others solutions within students' start-up projects connected with IT industry. We also plan to implement the project's results in other universities of Russia.

References

1. **Sorokin A.** Automation Beyond Web 2.0, *Informatics And Applications*, 2014, vol. 8, iss. 4, pp. 114—125.
2. **Device** democracy. Saving the future of the Internet of Things. IBM Global Business Services Executive Report. IBM Institute for Business Value. IBM Corporation. 2014. (DN: GBE03620USEN), available at: http://public.dhe.ibm.com/common/ssi/ecm/gb/en/gbe03620usen/GBE03620USEN.PDF
3. **Popov A. Yu., Chembaev V. D.** Opyt razrabotki otkazoustoychivogo kompleksa oblachnykh servisov dlya podderzhki obrazovatel'noy i nauchno-issledovatel'skoy deyatel'nosti MGTU im N. E.Baumana (Experience of development of failure-safe system of cloud services for educational and research activity supporting in Moscow State Technical University n.a. Bauman), *Inzhenernyj Vestnik*, 2014, no. 12, available at: http://ainjournal.ru/doc/745300.html (in Russian).
4. **Antonopoulos N., Gillam L. (Eds.).** *Cloud Computing: Principles, Systems and Applications*, London, Springer, 2017, 379 p.
5. **IBM** Intelligent Operations Center for Smarter Cities Administration Guide (2012). IBM Redbooks Solution Guide, available at: http://www.redbooks.ibm.com/redbooks/pdfs/sg248061.pdf
6. **Popov A., Proletarsky A., Belov S., Sorokin A.** Fast Prototyping of the Internet of Things solutions with IBM Cloud, *HICSS 50*, 3—7 January 2017, Hawaii, 2017, pp. 1064—1072, available at: http://hdl.handle.net/10125/41279
7. **Poseidon** project, available at: http://poseidonproject.org/

For citation:

Popov A. Yu., Belov S. A., Sorokin A. V. Cloud Based IT Learning Infrastructure to Support New Generation of Services, *Programmnaya Ingeneria,* 2018, vol. 9, no. 6, pp. 281—286.

DOI: 10.17587/prin.9.281-286

XX Всероссийская конференция

НАУЧНЫЙ СЕРВИС В СЕТИ ИНТЕРНЕТ

с 17 по 22 сентября 2018 г.

Конференцию проводит
Институт прикладной математики
им. М.В. Келдыша РАН

Конференция посвящена основным направлениям и тенденциям использования интернет-технологий в современных научных исследованиях. Основная цель конференции – предоставить возможность для обсуждения, апробации и обмена мнениями о наиболее значимых результатах, полученных ведущими российскими учеными за последнее время в данной области деятельности.

Конференция серии «Научный сервис в сети Интернет» проводится в двадцатый раз. В 2017 г. в ее работе приняли участие свыше 150 человек.

Тематика конференции

- Научные исследования и интернет, интернет-представительство научных организаций и проектов.
- Решение задач и обработка данных на суперкомпьютерах центров коллективного пользования.
- Интернет-проекты в области параллельных вычислений, математическое моделирование, вычислительные сервисы.
- Интернет-проекты для биомедицины.
- Модели и методы построения поисковых систем и систем навигации в интернете, технологии и системы распределенного хранения и обработки данных.
- Технологии и опыт построения информационных систем и баз данных, документации и результатов эксперимента на основе интернет-технологий.
- Цифровые библиотеки и библиографические базы, семантический веб, наукометрия в интернете.
- Онлайновая научная публикация, открытая наука, живая публикация, онлайновое рецензирование, мультимедийные иллюстрации.
- Популярный научный интернет, онлайновые энциклопедии, история науки в интернете.
- Интернет-активность ученого, персональная страница, профили ученого в библиографических базах, аттестация в интернете.
- Системное и инструментальное программное обеспечение, языки и модели программирования, формальные методы для интернет-технологий.

Конференция проводится с 17 по 22 сентября в пансионате «Моряк», расположенном в 20 километрах от Новороссийска в живописном месте на берегу Черного моря недалеко от поселка Дюрсо.

Сайт конференции: http://agora.guru.ru/abrau2018

11—13 октября 2018 г. в **МВДЦ "Сибирь",** г. Красноярск, состоится

itCOM 2018 —

специализированная выставка средств связи и телекоммуникаций, компьютеров, информационных и интернет-технологий

Основные тематические разделы выставки

Сети

- Оборудование и системы связи IT-систем и оборудования для корпоративных клиентов, предприятий, среднего и малого бизнеса
- Телерадиовещательная техника
- Телевизионные системы, оборудование видеоконференций, удаленного видеоконтроля и мониторинга объектов

Телекоммуникационные технологии

Сетевые компоненты и обеспечение

Приложения и Сервисы

- SaaS, "облачные" приложения, виртуализация
- IT-услуги, системная интеграция, IT-аутсорсинг, IT-консалтинг
- Сетевые решения
- Электронный документооборот
- Электронное правительство
- Система оперативного управления компанией
- Технологии мобильного позиционирования
- Мобильные и беспроводные сети связи общего и корпоративного пользования
- Комплексные системы автоматизации управления инфраструктурой предприятия, города, региона
- Технологии дистанционного обучения и подготовки кадров

Информационные системы, программные продукты

Системы защиты информации и управления данными

- Системы и средства защиты информации и персональных данных
- Антивирусная защита
- Управление данными и хранение информации

Сетевые компоненты и обеспечение

Подробности на сайте https://www.krasfair.ru/events/itCOM/

ООО "Издательство "Новые технологии". 107076, Москва, Стромынский пер., 4
Технический редактор *Е. М. Патрушева*. Корректор *Е. В. Комиссарова*

Сдано в набор 17.04.2018 г. Подписано в печать 22.05.2018 г. Формат 60×88 1/8. Заказ PI618
Цена свободная.

Оригинал-макет ООО "Адвансед солюшнз". Отпечатано в ООО "Адвансед солюшнз".
119071, г. Москва, Ленинский пр-т, д. 19, стр. 1. Сайт: **www.aov.ru**

Рисунок к статье Д. А. Юхимца, Э. Э. Юдинкова
«РАЗРАБОТКА ПРИКЛАДНОГО ПРОГРАММНОГО ИНТЕРФЕЙСА ДЛЯ УПРАВЛЕНИЯ РОБОТОМ-МАНИПУЛЯТОРОМ MITSUBISHI RV-2FB»

Рис. 3. Графический пользовательский интерфейс

Рисунки к статье А. А. Артемова
«ПРЕДИКТИВНАЯ ОЦЕНКА ВЕРХНЕЙ ГРАНИЦЫ ОШИБКИ ПРОГНОЗА МОДЕЛИ, ВОЗНИКАЮЩЕЙ ВСЛЕДСТВИЕ КОНЦЕПТУАЛЬНОГО СМЕЩЕНИЯ ДАННЫХ НА ПРИМЕРЕ МЕМ-ГРАММ-МОДЕЛИ»

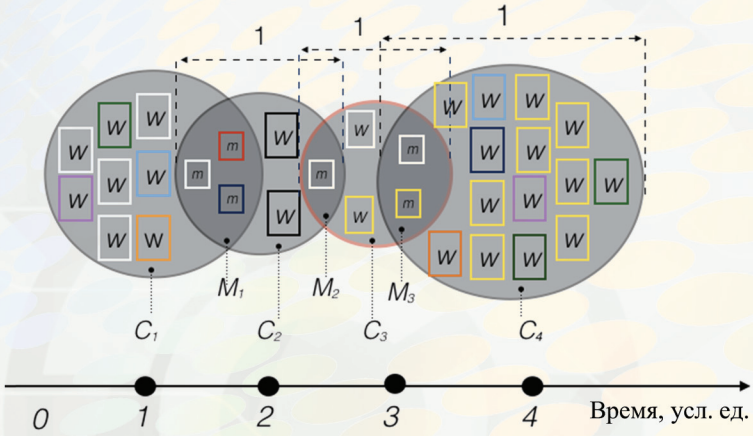


Рис. 2. Схематичное представление способа описания изменения содержания ИП в мем-грамм-модели. Элементы *m* множества *M* описывают наследственность ИП представленного мультимножества *C*, состоящего из элементов изменчивости *w*. Цветной рамкой проиллюстрирована общность элементов наследственности и изменчивости, совместно в каждый момент времени наследственность и изменчивость рассматриваются как единое целое мультимножество

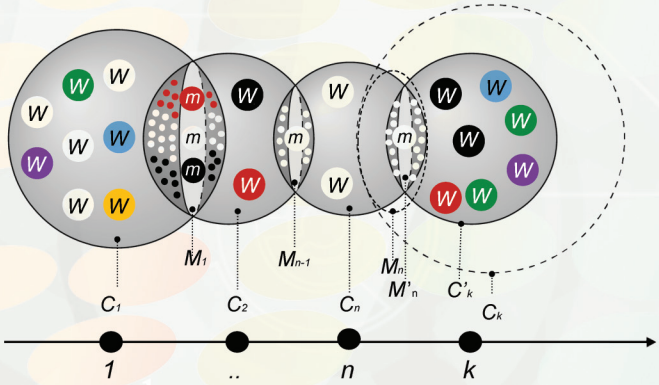


Рис. 4. Иллюстративное представление задачи: разными цветами представлены различные группы элементов наследственности и изменчивости

Издательство «НОВЫЕ ТЕХНОЛОГИИ»

выпускает *научно-технические журналы*

Теоретический и прикладной научно-технический журнал

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

В журнале освещаются состояние и тенденции развития основных направлений индустрии программного обеспечения, связанных с проектированием, конструированием, архитектурой, обеспечением качества и сопровождением жизненного цикла программного обеспечения, а также рассматриваются достижения в области создания и эксплуатации прикладных программно-информационных систем во всех областях челове ческой деятельности.

Подписные индексы по каталогам:
«Роспечать» – 22765; «Пресса России» – 39795

Ежемесячный теоретический и прикладной научно-технический журнал

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

В журнале освещаются современное состояние, тенденции и перспективы развития основных направлений в области разработки, производства и применения информационных технологий.

Подписные индексы по каталогам:
«Роспечать» – 72656;
«Пресса России» – 94033

Ежемесячный междисциплинарный теоретический и прикладной научно-технический журнал

НАНО- и МИКРОСИСТЕМНАЯ ТЕХНИКА

В журнале освещаются современное состояние, тенденции и перспективы развития нано- и микросистемной техники, рассматриваются вопрос разработки и внедрения нанои микросистем в различные области науки, технологии и производства.

Подписные индексы по каталогам:
«Роспечать» – 79493;
«Пресса России» – 27849

Ежемесячный теоретический и прикладной научно-технический журнал

МЕХАТРОНИКА, АВТОМАТИЗАЦИЯ, УПРАВЛЕНИЕ

В журнале освещаются достижения в области мехатроники, интегрирующей механику, электронику, автоматику и информатику в целях совершенствования технологий производства и создания техники новых поколений. Рассматриваются актуальные проблемы теории и практики автомати ческого и автоматизированного управления техническими объектами и технологическими процессами в промышленности, энергетике и на транспорте.

Подписные индексы по каталогам:
«Роспечать» – 79492;
«Пресса России» – 27848

Научно-практический и учебно-методический журнал

БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

В журнале освещаются достижения и перспективы в области исследований, обеспечения и совершенствования защиты человека от всех видов опасностей производственной и природной среды, их контроля, мониторинга, предотвращения, ликвидации последствий аварий и катастроф, образования в сфере безопасности жизнедеятельности.

Подписные индексы по каталогам:
«Роспечать» – 79963;
«Пресса России» – 94032

Адрес редакции журналов для авторов и подписчиков:
107076, Москва, Стромынский пер., 4. Издательство "НОВЫЕ ТЕХНОЛОГИИ".
Тел.: (499) 269-55-10, 269-53-97. Факс: (499) 269-55-10. E-mail: antonov@novtex.ru

ISSN 2220-3397. Программная инженерия. 2018. Том 9. № 6. 241–288. Индексы: «Роспечать» – 22765, «Пресса России» – 39795