ВЕСТНИК
НОВОСИБИРСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

Научный журнал
Основан в ноябре 1999 года

Серия: Информационные технологии 2021. Том 19, № 2

СОДЕРЖАНИЕ

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2021. Том 19, № 2
Vestnik NSU. Series: Information Technologies, 2021, vol. 19, no. 2

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2021. Том 19, № 2
Vestnik NSU. Series: Information Technologies, 2021, vol. 19, no. 2

V E S T N I K
NOVOSIBIRSK STATE UNIVERSITY

Scientific Journal
Since 1999, November
In Russian

Series: Information Technologies 2021. Volume 19, № 2

CONTENTS

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2021. Том 19, № 2
Vestnik NSU. Series: Information Technologies, 2021, vol. 19, no. 2

The journal is published quarterly in Russian since 1999
by Novosibirsk State University Press

The address for correspondence

Faculty of Information Technologies, Novosibirsk State University

1 Pirogov Street, Novosibirsk, 630090, Russia

Tel. +7 (383) 363 42 46

E-mail address: inftech@vestnik.nsu.ru

On-line version: http://elibrary.ru

УДК 004.912 + 004.8
DOI 10.25205/1818-7900-2021-19-2-5-16

Метод автоматического извлечения терминов из научных статей на основе слабо контролируемого обучения

Е. П. Бручес, Т. В. Батура

Институт систем информатики им. А. П. Ершова СО РАН
Новосибирск, Россия

Новосибирский государственный университет
Новосибирск, Россия

Аннотация

Описывается метод извлечения научных терминов из текстов на русском языке, основанный на слабо контролируемом обучении (weakly supervised learning). Особенность данного метода заключается в том, что для него не нужны размеченные вручную данные, что является очень актуальным. Для реализации метода мы собрали в полуавтоматическом режиме словарь терминов, затем автоматически разметили тексты научных статей этими терминами. Полученные тексты мы использовали для обучения модели. Затем этой моделью были автоматически размечены другие тексты. Вторая модель была обучена на объединении текстов, размеченных словарем и первой моделью. Результаты показали, что добавление данных, полученных даже автоматической разметкой, улучшает качество извлечения терминов из текстов.

Ключевые слова

извлечение терминов, нейросетевые модели языка, словарный метод, слабо контролируемое обучение, распознавание сущностей, обработка текстов

Благодарности

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 19-07-01134

Для цитирования

Бручес Е. П., Батура Т. В. Метод автоматического извлечения терминов из научных статей на основе слабо контролируемого обучения // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 5–16. DOI 10.25205/1818-7900-2021-19-2-5-16

Method for Automatic Term Extraction from Scientific Articles Based on Weak Supervision

E. P. Bruches, T.V. Batura

A. P. Ershov Institute of Informatics Systems SB RAS
Novosibirsk, Russian Federation

Novosibirsk State University
Novosibirsk, Russian Federation

Abstract

We propose a method for scientific terms extraction from the texts in Russian based on weakly supervised learning. This approach doesn't require a large amount of hand-labeled data. To implement this method we collected a list of terms in a semi-automatic way and then annotated texts of scientific articles with these terms. These texts we used to train a model. Then we used predictions of this model on another part of the text collection to extend the train set. The second model was trained on both text collections: annotated with a dictionary and by a second model. Obtained results showed that giving additional data, annotated even in an automatic way, improves the quality of scientific terms extraction.

Keywords

term extraction, neural network language models, dictionary approach, weakly supervised learning, entity recognition, text processing

Acknowledgements

The study was funded by RFBR according to the research project N 19-07-01134

For citation

Bruches E. P., Batura T. V. Method for Automatic Term Extraction from Scientific Articles Based on Weak Supervision. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 5–16. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-5-16

Введение

В настоящее время всё большую актуальность приобретает автоматическая обработка текстов научных статей, которая включает в себя многие задачи: извлечение терминов, извлечение отношений между терминами, автореферирование и др. Это связано с ростом числа публикуемых работ. Чтение и анализ публикаций в поисках нужной информации становится всё более затратной по времени и ресурсам задачей. В связи с этим в последнее время появляются различные системы автоматического анализа научных статей, например ScholarPhi [1].

В данной статье мы предлагаем метод автоматического извлечения терминов из текстов научных статей на русского языке. Эта задача осложняется тем, что для русского языка не только нет размеченных данных в достаточном количестве, но также довольно проблематично получить неразмеченные тексты статей. Мы разработали метод, не требующий размеченных вручную данных. При этом достигнутые результаты показывают, что данный модуль может быть использован в последующих задачах, требующих информацию о научных терминах.

В данной работе под термином понимается слово или словосочетание, являющееся названием некоторого понятия какой-нибудь области науки, техники, искусства и др. [2]. Общая идея, которая лежит в основе традиционных подходов, состоит в том, что автоматическое извлечение терминов происходит в два этапа: на первом этапе из текстов извлекаются *n*-граммы слов, которые потенциально могут быть терминами, а на втором этапе выполняется классификация, в результате которой принимается решение, является ли данная фраза термином. Алгоритмы, архитектура которых соответствует этой идее, можно также разделить на несколько групп.

Первая группа предполагает использование правил для выделения из текстов фраз, которые являются терминами. Например, в работе [3] предлагается использование словарей и информации о синтаксической структуре предложения для извлечения многословных терминов.

Другую группу составляют методы, в основе которых лежит использование алгоритмов машинного обучения с вручную извлеченными признаками. Так, в работе [4] описывается алгоритм извлечения как однословных, так и многословных терминов: на первом этапе из текстов извлекаются *n*-граммы слов, которые потенциально могут быть терминами, а затем на основании различных входных признаков алгоритм определяет, является ли *n*-грамма термином. В статье [5] авторы используют несколько групп признаков для извлечения терминов: лингвистические (части речи, главное слово фразы, количество имен существительных во фразе и др.), статистические (длина фразы, TF, IDF, TF-IDF и др.) и гибридные признаки (например, частота встречаемости фразы в корпусах обычных и научных текстов). Также было исследовано применение алгоритма PageRank для более точной классификации [6]. В работе [7] предлагается использовать признаки, основанные на информации из Википедии.

В третью группу входят методы глубокого обучения. В работе [8] исследуется проблема отсутствия достаточного количества данных. Для этого авторы на ограниченном количестве

данных обучают две модели (CNN и LSTM), которые на вход принимают векторные представления слов фразы, а на выходе определяют, является данная фраза термином или нет. Затем этими моделями размечается новая порция данных, которая добавляется в обучающую выборку, и процесс обучения повторяется еще раз.

В работе [9] предлагается архитектура, состоящая из этапов, отличных от тех, что были описаны: на первом шаге классификатор определяет, содержит ли входное предложение термины; если содержит, то на втором этапе происходит непосредственно нахождение терминов в предложении.

Другая общая идея – рассматривать задачу извлечения терминов как задачу сопоставления последовательностей входных токенов с последовательностями меток из заранее определенного множества (sequence labelling task), т. е. для каждого токена в тексте требуется определить его класс (является он термином или нет). Таким образом, решение задачи осуществляется в один этап. Как правило, при таком подходе используется разметка сущностей в формате BIO (BILOU и др.). Большим преимуществом данного подхода является то, что во внимание принимается контекст (как семантический, так и синтаксический) употребления конкретной фразы, что составляет один из ключевых признаков для нахождения терминов в тексте. Так, в работе [10] исследуются различные архитектуры и векторные представления слов при решении задачи sequence labelling.

Еще одна идея, которая отличается от описанных выше, состоит в использовании методов тематического моделирования для извлечения терминов. В статье [11] описывается попытка применения различных методов тематического моделирования для улучшения нахождения однословных терминов: невероятностные (разные методы кластеризации – K-means, NFM и др.) и вероятностные (в качестве метода такой группы был выбран алгоритм LDA).

Алгоритм извлечения терминов

Ввиду отсутствия достаточного количества размеченных данных для задачи извлечения терминов для русского языка мы приняли решение использовать подход псевдоразметки (pseudo-labelling). Он заключается в том, чтобы обучить модель на небольшом количестве размеченных данных, а затем разметить полученной моделью некоторое количество новых текстов, добавить их к обучающему множеству и обучить вторую модель.

Таким образом, алгоритм получения модели для извлечения терминов состоит из следующих шагов:

1) получить размеченный корпус для первой итерации обучения модели с помощью словарного подхода;

2) обучить модель на полученном корпусе из п. 1;

3) разметить новые тексты и тексты из п. 1 моделью, полученной в результате выполнения п. 2, и словарным подходом;

4) обучить модель на полученном корпусе текстов из п. 3.

Схема алгоритма представлена на рис. 1.

Рассмотрим каждый из шагов более детально.

Получение размеченного корпуса для первой итерации обучения модели

Идея этого подхода состоит в использовании заранее составленного словаря терминов. Словарь терминов набирался в полуавтоматическом режиме двумя способами.

1. Из текстов научных статей были автоматически извлечены 2-, 3- и 4-граммы слов, отсортированные по значению tf-idf, затем из них вручную были выбраны те фразы, которые потенциально могут быть терминами.

2. Из заголовков статей из Википедии, входящих в подграф категории «Наука», были вручную выбраны те слова и фразы, которые потенциально могут быть терминами.

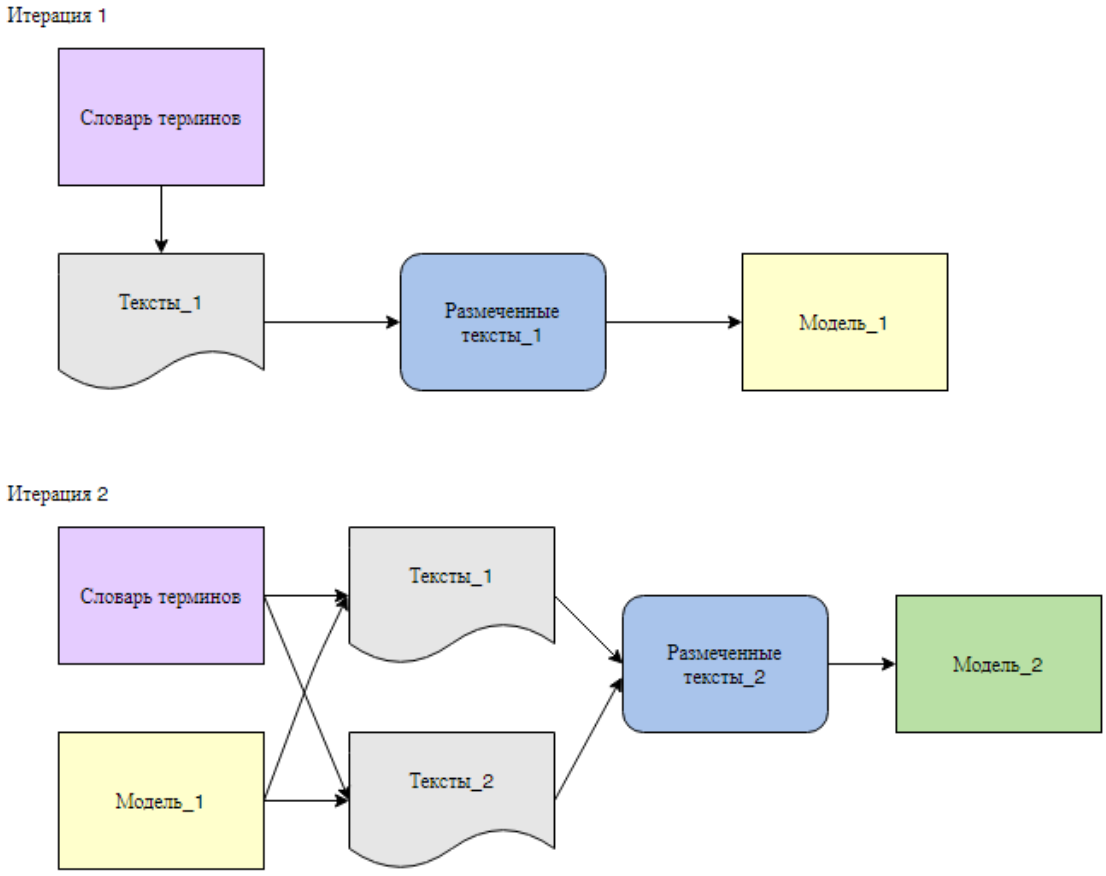


Рис. 1. Схема алгоритма
Fig. 1. Algorithm

Таким образом был получен словарь из 17 252 терминов:

- 1-граммы (6 509 терминов): *этнолингвистика*, *равносоставленность*, *пластида* и др.;
- 2-граммы (6 785 терминов): *нокдаун гена*, *математическое ожидание* и др.
- 3-граммы (2 322 терминов): *уравнение переноса излучений*, *метод анализа иерархий* и др.;
- 4-граммы (1 098 терминов): *теорема Бертрана о выборах*, *стабилизированный метод бисопряжённых градиентов* и др.;
- 5-граммы (348 терминов): *теория реакционной способности химических соединений* и др.;
- 6-граммы (120 терминов): *теорема Римана об условно сходящихся рядах*, *задача о назначении минимального количества исполнителей* и др.;
- 7-граммы (40 терминов): *теорема о свойстве Дарбу для непрерывной функции* и др.;
- 8-граммы (22 термина): *теорема Пуанкаре о разложении интегралов по малому параметру* и др.;
- 9-граммы (7 терминов): *теорема Стоуна о группах унитарных операторов в гильбертовом пространстве* и др.;
- 12-граммы (1 термин): *автоматический выключатель*, *управляемый дифференциальным током*, *со встроенной защитой от сверхтока*.

Основная сложность составления словаря терминов заключается в том, что без контекста сложно понять, является фраза термином или нет. Более того, в разных контекстах одна и та же фраза может быть как термином, так и нет; например, *модель*, *текст*, *язык* и др.

Таким образом, было размечено 1 118 текстов научных статей из журнала «Программные системы и продукты», которые использовались в качестве обучающего множества для модели в первой итерации (Bert-iter_1). Полученный словарь терминов находится в открытом доступе [1].

Следует отметить, что этап создания словаря может быть заменен на этап пополнения уже имеющегося словаря терминов выбранной предметной области (при наличии подходящего). Для экспериментов мы выбрали область компьютерных технологий как пример быстро меняющейся отрасли, в которой в настоящий момент не вся терминология является устоявшейся: одни термины появляются, другие быстро устаревают. В таких условиях довольно сложно составлять словари терминов и поддерживать их в актуальном состоянии.

Получение размеченного корпуса для второй итерации обучения модели

На следующем шаге мы взяли 808 аннотаций научных статей из журналов «Cloud of science», «Программные системы и вычислительные методы», «Информационно-управляющие системы» и разметили их комбинированной моделью. Данная модель (Bert-iter_2) представляет собой комбинацию словарного метода, с помощью которого были размечены тексты в предыдущем пункте, и модели, полученной после первой итерации обучения.

Обучающим множеством для модели второй итерации стало объединение текстов первой итерации и новых размеченных текстов.

Основная гипотеза при использовании предсказаний модели для разметки текстов состоит в том, что обобщающая способность модели позволяет не только выделять термины, которые уже содержатся в словаре, но находить новые паттерны и, соответственно, новые слова и фразы, которые являются терминами.

Описание модели

В обеих итерациях мы использовали одну и ту же архитектуру нейронной сети. Векторные представления слов для входных текстов мы получали, используя предобученную модель bert-base-multilingual-cased [2]. Затем идут слой двунаправленной LSTM и два полносвязных слоя. Архитектура модели показана на рис. 2.

На вход модели подается токенизированный текст (входные тексты предварительно никак не обрабатываются). Выход модели представляет собой последовательность предсказанных классов для соответствующих токенов. В данной задаче используется набор из трех классов: «B-TERM» (обозначает первый токен в термине, *beginning*)), «I-TERM» (обозначает не первый токен в термине, *inside*), «O» (обозначает токен, который не входит в состав термина, *outside*).

Для улучшения качества извлечения терминов, мы также написали валидацию эвристиками. Ниже приводится описание используемых эвристик.

Описание эвристик

Эвристика 1: если токен (1) распознан как термин и является именем существительным или именем прилагательным, и следующий токен (2) распознан как не термин и имеет форму родительного падежа, то токену (2) присваивается тэг «I-TERM». Примеры последовательностей: *методы сжатия (1) данных (2)*, *реляционных баз (1) данных (2)*, *системы (1) проверки (2) заданий* и др.

[1] https://github.com/iis-research-team/ner-rc-russian/tree/master/dict_extractor
[2] https://huggingface.co/transformers/pretrained_models.html

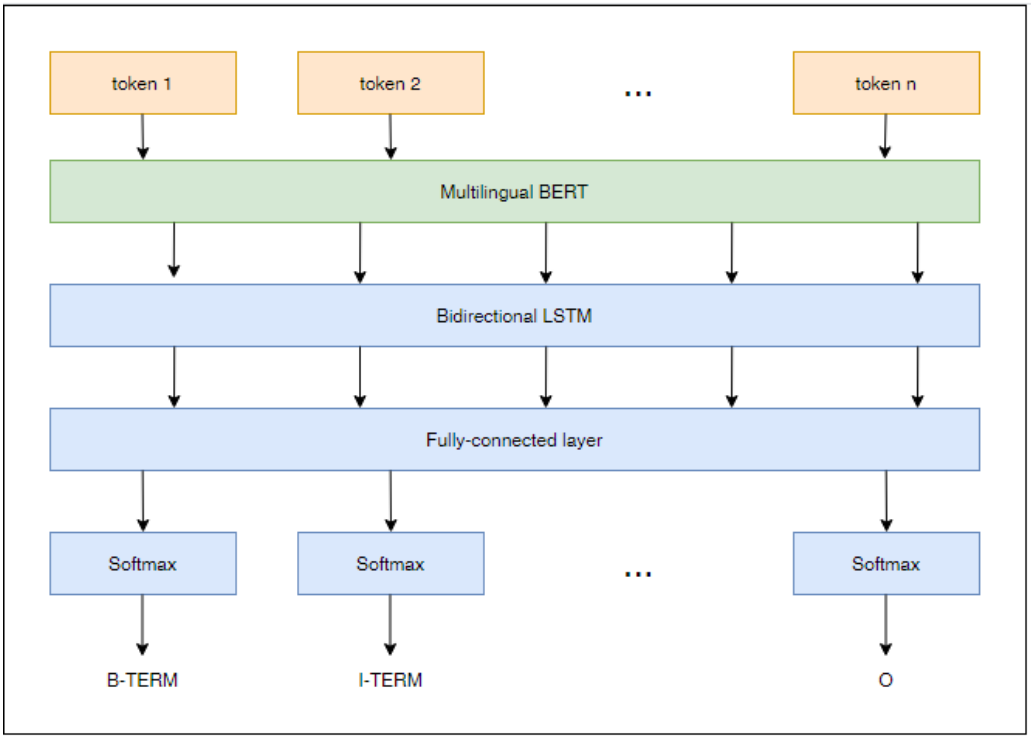


Рис. 2. Архитектура нейронной сети
Fig. 2. Neural network architecture

Эвристика 2: если токен (1) распознан как «B-TERM» и является именем прилагательным, и токен (2) распознан как «B-TERM» и является именем существительным, то меняем тэг токена (2) на «I-TERM». Примеры последовательностей: *в учебном (1) процессе (2)*, *нечёткая (1) модель (2)*, *физические (1) величины (2)* и др.

Эвристика 3: если последний токен (1) в цепочке токенов, образующих термин, имеет часть речи имя прилагательное, а следующий за ним токен имеет часть речи имя существительное (2), то токену (2) присваивается тэг «I-TERM». Примеры последовательностей: *возможности мобильных (1) устройств (2)* и др.

Эвристика 4: если токен (1) входит в состав термина, а следующий за ним токен состоит только из латинских символов, то включаем его в состав термина. Пример последовательностей: *в пакете (1) MOST (2)*, *по базе данных (1) Cistrome (2)* и др.

Также мы в явном виде запрещаем помечать тэгами терминов следующие классы токенов:

1) знаки пунктуации: «.», «,», «:», «;»;

2) предлоги и союзы, если они имеют тэг «B-TERM» (мы допускаем появление предлогов и союзов в составе термина, но не допускаем того, что данные части речи начинают термин);

3) однозначные глаголы и деепричастия (под однозначными подразумевается, что токены не имеют морфологической омонимии).

Анализ результатов

Полученные на каждом шаге модели мы оценивали на корпусе RuSERRC [12], который не использовался в процессе обучения.

Для оценки качества моделей мы использовали стандартные метрики классификации: точность, полноту и F-меру. Точность – это доля верно извлеченных терминов относительно

всех терминов, которые извлекла модель. Полнота – это доля верно найденных моделью терминов относительно всех терминов в тестовой выборке. F-мера представляет собой гармоническое среднее между точностью и полнотой (в табл. 1 приведена взвешенная F-мера).

При этом мы рассматривали несколько вариантов того, что имеется в виду под словом «термин».

В первом случае под термином понимается вся последовательность токенов, входящая в состав термина. Если хотя бы один токен распознан неверно, то считается, что весь термин распознан неверно. В табл. 1 такая метрика указана как «Полное совпадение».

Во втором случае под термином понимается токен, тэг которого принадлежит множеству {“B-TERM”, “I-TERM”}. Ввиду большой неоднозначности определения границ терминов, которая присутствовала также при разметке корпуса асессорами, данная метрика видится нам релевантной – она показывает, насколько хорошо модель способна распознать фразы, которые могут быть терминами, без указания точных границ. В табл. 1 такая метрика указана как «Частичное совпадение».

Для сравнения мы взяли алгоритм извлечения ключевых слов RAKE, применение которого к этому корпусу описано в статье [13]. Rapid automatic keyword extraction (RAKE) – алгоритм, предназначенный для автоматического извлечения ключевых слов [14]. Сначала применяется список стоп-слов и разделителей для выделения многословных терминов, после чего используется статистическая информация: для каждого слова из ключевых фраз-кандидатов оценивается частота, с которой оно встречалось, и количество связей между этим словом и остальными. На основании этих двух величин вычисляется вес ключевой фразы, и все фразы сортируются по весам, а наиболее вероятные ключевые фразы получают максимальный вес. Оригинальный алгоритм в качестве ключевых слов выделяет также и глагольные формы, но так как в данной задаче рассматриваются только именные группы, то в статье описан оптимизированный алгоритм RAKE, в котором на этапе предобработки текста удаляются все глагольные формы. Полученные метрики приведены в табл. 1.

Таблица 1

Метрики извлечения терминов

Table 1

Term Extraction Metrics

Метод	Полное совпадение			Частичное совпадение		
	Точность	Полнота	F-мера	Точность	Полнота	F-мера
Bert-iter_1	0,22	0,19	0,20	0,76	0,71	0,68
Bert-iter_1 + эвристики	0,39	0,28	0,33	0,76	0,75	0,74
Bert-iter_2	0,30	0,25	0,28	0,77	0,75	0,74
Bert-iter_2 + эвристики	0,40	0,29	0,34	0,77	0,77	0,76
Bert-iter_2 + эвристики + словарь	0,39	0,31	0,35	**0,78**	**0,78**	**0,77**
Rake	0,36	0,28	0,32	0,62	0,63	0,63
Оптимизированный Rake	**0,44**	**0,35**	**0,39**	0,65	0,57	0,61

Относительно низкие метрики во многом связаны с различием разметки обучающего множества и золотого стандарта. В силу того что обучающее множество было получено с помощью автоматической разметки терминов из словаря, последовательность токенов, являющаяся термином, не претерпевала никаких изменений. В реальных текстах термин может быть «разорван», содержать синонимы, сокращения, пунктуационные знаки или вовсе быть неполным.

Если проанализировать метрики частичного совпадения токенов, то видно, что модель способна находить места в тексте, в которых может быть термин, без точного определения границ термина. Учитывая, что задача определения границ термина является достаточно сложной даже для человека (что, например, показывает метрика согласованности асессоров при разметке сущностей в корпусе), то полученные метрики кажутся нам достаточными для использования данного подхода при решении других задач (например, задачи связывания именованных сущностей или классификации отношений между сущностями).

Применение модели к текстам другой предметной области

Для этого эксперимента мы использовали размеченные тексты из корпуса RuREBus [15], который содержит программы стратегического развития, предоставленные Минэкономразвития РФ. Так как мы работаем с текстами из области математики и информационных технологий, то область экономики как раз подходит для проверки способности модели к обобщению. Более того, жанры текстов также различаются: модель была обучена на текстах научных статей, в то время как в корпусе RuREBus содержатся тексты различных экономических документов. В разметке этого корпуса используется 8 типов именованных сущностей:

1) метрика – индикатор или объект, на основании которого производится сравнение (например, *уровень рождаемости*, *экономический рост*);

2) экономика – экономическая сущность или объект инфраструктуры (например, *ПАО Газпром*, *библиотечные и музейные фонды*);

3) институт – институты, структуры и организации (например, *Центр занятости молодёжи*, *системы дорог*);

4) бинарный – бинарная характеристика или единичное действие (например, *модернизация*, *функционирует*);

5) сравнение – сравнительная характеристика (например, *рост*, *негативная динамика*);

6) качество – качественная характеристика (например, *эффективный*, *безопасный*);

7) социальный – социальный объект (например, *научный и образовательный потенциал*, *досуг*);

8) деятельность – деятельность, события (например, *реставрационные работы*, *ярмарка выходного дня*).

Из приведенных примеров видно, что под сущностями авторы RuREBus понимают не только именные группы, но и глаголы и глагольные группы, и отдельные прилагательные. Такой принцип выделения именованных сущностей расходится с тем, которому мы следуем в данной работе, – под именованными сущностями мы подразумеваем только существительные и именные группы. Поэтому нам кажется некорректным непосредственное сравнение метрик. Тем не менее мы можем проанализировать результаты и сделать некоторые выводы.

В табл. 2 представлены метрики полного и частичного совпадений, полученные нашей моделью на корпусе RuREBus. Очевидно, что модель плохо находит границы терминов, но способна находить слова и фразы, которые входят в состав термина.

Таблица 2

Метрики на корпусе RuREBus

Table 2

Metrics for RuREBus

Метод	Полное совпадение			Частичное совпадение		
	Точность	Полнота	F-мера	Точность	Полнота	F-мера
Bert-iter_2 + эвристики + словарь	0,17	0,13	0,15	0,68	0,36	0,47

Таблица 3

Примеры выделения терминов из текстов RuREBus

Table 3

Examples of terms extraction from RuREBus texts

№ п/п	Текст
1	*Повышение **[доступности транспортных услуг]** для **[населения]**.*
2	*Организация **[временного трудоустройства отдельных категорий безработных граждан]**.*
3	***[Право] [граждан]** на благоприятную **[среду жизнедеятельности]** закреплено в основном **[законе государства] - [Конституции Российской Федерации]**.*
4	*Контроль за исполнением **[постановления]** возложить на **[первого заместителя руководителя администрации]** МР "Усть-Вымский" Карпову А.Д.*
5	***[Оценка эффективности рисков] - [риски]** низкие.*
6	***[Деятельность учреждений культуры]** и искусства является одной из важнейших составляющих современной культурной жизни.*
7	***[ПОСТАНОВЛЕНИЕ]** об утверждении муниципальной программы "**[Поддержка сельскохозяйственных товаропроизводителей]** и **[создание условий]** для **[развития сферы заготовки]** и **[переработки дикорастущего сырья Верхнекетского района]** на 2016-2021 годы".*
8	***[Паспорт] [муниципальной программы]** городского округа Кашира "**[Спорт городского округа]** Кашира" на 2017-2021 годы.*
9	*Приложение к **[постановлению администрации муниципального имущества]***
10	*За этот период было установлено и заменено 366 дорожных **[знаков]** и 43 сигнальных столбика на **[железнодорожных переездах]**.*

В табл. 3 приведены примеры полученной нами разметки предложений из корпуса RuREBus с помощью предложенного подхода – в квадратных скобках заключены выделенные сущности. Видно, что в тексте автоматически выделяются последовательности токенов, которые являются сущностями в данном контексте. Таким образом, можно заключить, что данный метод потенциально может применяться к текстам из различных предметных областей и давать приемлемые результаты в пределах заданного языка (в нашем случае русского) без дополнительных затрат.

Заключение

В данной работе описан метод автоматического извлечения терминов из научных текстов на русском языке. Метод основан на слабо контролируемом обучении и позволяет решать задачу без размеченных вручную данных, что является особенно актуальным на сегодняшний день. Полученные результаты показали эффективность использования дополнительных данных, размеченных даже автоматически. Кроме того, анализ применения данной модели к текстам другой области знаний показал, что модель способна к обобщению.

Список литературы

1. **Head A.**, **Lo K.**, **Kang D.**, **Fok R.**, **Skjonsberg S.**, **Weld D. S.**, **Hearst M. A.** Augmenting Scientific Papers with Just-in-Time, Position-Sensitive Definitions of Terms and Symbols. ArXiv: 2009.14237. 2021.

2. **Лопатин В. В.**, **Лопатина Л. Е.** Русский толковый словарь: около 35 000 слов. М.: Русский язык, 1997. 832 с.
3. **Stanković R.**, **Krstev C.**, **Obradović I.**, **Lazić B.**, **Trtovac A.** Rule-based Automatic Multi-word Term Extraction and Lemmatization. In: Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16). 2016, p. 507–514.
4. **Yuan Y.**, **Gao J.**, **Zhang Y.** Supervised Learning for Robust Term Extraction. In: Proceedings of 2017 International Conference on Asian Language Processing (IALP). 2017, p. 302–305. DOI 10.1109/IALP.2017.8300603
5. **Conrado M.**, **Pardo T.**, **Rezende S. O.** A Machine Learning Approach to Automatic Term Extraction using a Rich Feature Set. In: Proceedings of the NAACL HLT 2013 Student Research Workshop. Atlanta, Georgia, 2013, p. 16–23.
6. **Zhang Z.**, **Gao J.**, **Ciravegna F.** SemRe-Rank: Improving Automatic Term Extraction by Incorporating Semantic Relatedness with Personalised PageRank. *ACM Transactions on Knowledge Discovery from Data* (*TKDD*), 2018, vol. 12, no. 5, p. 1–41.
7. **Bilu Y.**, **Gretz Sh.**, **Cohen E.**, **Slonim N.** What if we had no Wikipedia? Domain-independent Term Extraction from a Large News Corpus. arXiv: 2009.08240. 2020.
8. **Wang R.**, **Liu W.**, **McDonald C.** Featureless Domain-Specific Term Extraction with Minimal Labelled Data. In: Proceedings of Australasian Language Technology Association Workshop, 2016, p. 103−112.
9. **Hossari M.**, **Dev S.**, **Kelleher J. D.** TEST: A Terminology Extraction System for Technology Related Terms. In: Proceedings of the 2019 11th International Conference on Computer and Automation Engineering, 2019, p. 78–81. DOI 10.1145/3313991.3314006
10. **Kucza M.**, **Niehues J.**, **Zenkel T.**, **Waibel A.**, **Stüker S.** Term Extraction via Neural Sequence Labeling a Comparative Evaluation of Strategies Using Recurrent Neural Networks. In: Proceedings of Interspeech 2018. 2018. p. 2072-2076.
11. **Bolshakova E.**, **Loukachevitch N.**, **Nokel M.** Topic Models Can Improve Domain Term Extraction. In:European Conference on Information Retrieval (ECIR 2013). Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2013, vol. 7814, p. 684–687.
12. **Bruches E.**, **Pauls A.**, **Batura T.**, **Isachenko V.** Entity Recognition and Relation Extraction from Scientific and Technical Texts in Russian. In: Science and Artificial Intelligence conference, 2020, p. 41–45. DOI 10.1109/S.A.I.ence50533.2020.9303196
13. **Бручес Е. П.**, **Паульс А. Е.**, **Батура Т. В.**, **Исаченко В. В.**, **Щербатов Д. Р.** Семантический анализ научных текстов: опыт создания корпуса и построения языковых моделей // Программные продукты и системы, 2020, 18 с.
14. **Rose S.**, **Engel D.**, **Cramer N.**, **Cowley W.** Automatic Keyword Extraction from Individual Documents. *Text Mining: Theory and Applications*, 2010, vol. 1, p. 1–20. DOI 10.1002/9780470689646.ch1
15. **Ivanin V.**, **Artemova E.**, **Batura T.**, **Ivanov V.**, **Sarkisyan V.**, **Tutubalina E.**, **Smurov I.** RUREBUS-2020 Shared Task: Russian Relation Extraction for Business. In: Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialog", 2020, p. 416–431. DOI 10.28995/2075-7182-2020-19-416-431

References

1. **Head A.**, **Lo K.**, **Kang D.**, **Fok R.**, **Skjonsberg S.**, **Weld D. S.**, **Hearst M. A.** Augmenting Scientific Papers with Just-in-Time, Position-Sensitive Definitions of Terms and Symbols. ArXiv: 2009.14237. 2021.
2. **Lopatin V. V.**, **Lopatina L. E.** Russian Explanatory Dictionary. Moscow, 1997, 832 p. (in Russ.)

3. **Stanković R.**, **Krstev C.**, **Obradović I.**, **Lazić B.**, **Trtovac A.** Rule-based Automatic Multi-word Term Extraction and Lemmatization. In: Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16). 2016, p. 507–514.
4. **Yuan Y.**, **Gao J.**, **Zhang Y.** Supervised Learning for Robust Term Extraction. In: Proceedings of 2017 International Conference on Asian Language Processing (IALP). 2017, p. 302–305. DOI 10.1109/IALP.2017.8300603
5. **Conrado M.**, **Pardo T.**, **Rezende S. O.** A Machine Learning Approach to Automatic Term Extraction using a Rich Feature Set. In: Proceedings of the NAACL HLT 2013 Student Research Workshop. Atlanta, Georgia, 2013, p. 16–23.
6. **Zhang Z.**, **Gao J.**, **Ciravegna F.** SemRe-Rank: Improving Automatic Term Extraction by Incorporating Semantic Relatedness with Personalised PageRank. *ACM Transactions on Knowledge Discovery from Data* (*TKDD*), 2018, vol. 12, no. 5, p. 1–41.
7. **Bilu Y.**, **Gretz Sh.**, **Cohen E.**, **Slonim N.** What if we had no Wikipedia? Domain-independent Term Extraction from a Large News Corpus. arXiv: 2009.08240. 2020.
8. **Wang R.**, **Liu W.**, **McDonald C.** Featureless Domain-Specific Term Extraction with Minimal Labelled Data. In: Proceedings of Australasian Language Technology Association Workshop, 2016, p. 103−112.
9. **Hossari M.**, **Dev S.**, **Kelleher J. D.** TEST: A Terminology Extraction System for Technology Related Terms. In: Proceedings of the 2019 11th International Conference on Computer and Automation Engineering, 2019, p. 78–81. DOI 10.1145/3313991.3314006
10. **Kucza M.**, **Niehues J.**, **Zenkel T.**, **Waibel A.**, **Stüker S.** Term Extraction via Neural Sequence Labeling a Comparative Evaluation of Strategies Using Recurrent Neural Networks. In: Proceedings of Interspeech 2018. 2018. p. 2072-2076.
11. **Bolshakova E.**, **Loukachevitch N.**, **Nokel M.** Topic Models Can Improve Domain Term Extraction. In:European Conference on Information Retrieval (ECIR 2013). Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2013, vol. 7814, p. 684–687.
12. **Bruches E.**, **Pauls A.**, **Batura T.**, **Isachenko V.** Entity Recognition and Relation Extraction from Scientific and Technical Texts in Russian. In: Science and Artificial Intelligence conference, 2020, p. 41–45. DOI 10.1109/S.A.I.ence50533.2020.9303196
13. **Bruches E.**, **Pauls A.**, **Batura T.**, **Isachenko V.**, **Shcherbatov D.** Semantic Analysis of Scientific Texts: Experience in Creating a Corpus and Building Language Models. *Software & Systems*, 2020, 18 p. (in Russ.)
14. **Rose S.**, **Engel D.**, **Cramer N.**, **Cowley W.** Automatic Keyword Extraction from Individual Documents. *Text Mining: Theory and Applications*, 2010, vol. 1, p. 1–20. DOI 10.1002/9780470689646.ch1
15. **Ivanin V.**, **Artemova E.**, **Batura T.**, **Ivanov V.**, **Sarkisyan V.**, **Tutubalina E.**, **Smurov I.** RUREBUS-2020 Shared Task: Russian Relation Extraction for Business. In: Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialog", 2020, p. 416–431. DOI 10.28995/2075-7182-2020-19-416-431

Материал поступил в редколлегию
Received
15.02.2021

Сведения об авторах

Батура Татьяна Викторовна, кандидат физико-математических наук, старший научный сотрудник, Институт систем информатики им. А. П. Ершова СО РАН (Новосибирск, Россия); доцент, Новосибирский государственный университет (Новосибирск, Россия)

tatiana.v.batura@gmail.com
ORCID 0000-0003-4333-7888

Бручес Елена Павловна, аспирант, Институт систем информатики им. А. П. Ершова СО РАН (Новосибирск, Россия); ассистент, Новосибирский государственный университет (Новосибирск, Россия)

bruches@bk.ru

Information about the Authors

Tatiana V. Batura, PhD in Physics and Mathematics, Senior Researcher, A. P. Ershov Institute of Informatics Systems SB RAS (Novosibirsk, Russian Federation); Associate Professor, Novosibirsk State University (Novosibirsk, Russian Federation)

tatiana.v.batura@gmail.com
ORCID 0000-0003-4333-7888

Elena P. Bruches, PhD student, A. P. Ershov Institute of Informatics Systems SB RAS (Novosibirsk, Russian Federation); Assistant, Novosibirsk State University (Novosibirsk, Russian Federation)

bruches@bk.ru

УДК 004.8, 004.822
DOI 10.25205/1818-7900-2021-19-2-17-28

Методики анализа киберситуационной осведомленности об энергетическом объекте

Д. А. Гаськова, А. Г. Массель

Институт систем энергетики им. Л. А. Мелентьева СО РАН
Иркутск, Россия

Аннотация

Предлагается осуществлять анализ киберситуационной осведомленности об энергетическом объекте в три этапа: анализ киберугроз энергетической инфраструктуры; моделирование сценариев экстремальных ситуаций в энергетике, вызванных реализацией киберугроз; оценка рисков нарушения кибербезопасности энергетической инфраструктуры. Представлены три методики, соответствующие каждому этапу. В рамках методического аппарата авторы предлагают применять семантические методы для анализа влияния киберугроз на объекты энергетики с учетом энергетической безопасности, которые показывают свою эффективность в условиях отсутствия или неполноты данных при моделировании поведения систем, которое не поддается формальному описанию или достаточно точному прогнозированию. Представлен подход к анализу киберситуационной осведомленности об энергетических объектах как синтез исследований кибербезопасности и ситуационной осведомленности, отличающийся использованием семантического моделирования.

Ключевые слова

киберситуационная осведомленность, критические ситуации в энергетике, семантические методы моделирования, киберугрозы

Благодарности

Результаты получены в рамках выполнения проекта по госзаданию ИСЭМ СО РАН FWEU-2021-0007 № АААА-А21-121012090007-7, отдельные аспекты прорабатывались в рамках проектов, поддержанных грантами РФФИ № 19-07-00351, № 20-010-00204, Бел_мол_а № 19-57-04003

Для цитирования

Гаськова Д. А., *Массель А. Г.* Методики анализа киберситуационной осведомленности об энергетическом объекте // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 17–28. DOI 10.25205/1818-7900-2021-19-2-17-28

The Method of Cyber Awareness Analysis of an Energy Facility

D. A. Gaskova, A. G. Massel

Melentiev Energy Systems Institute SB RAS
Irkutsk, Russian Federation

Abstract

The article proposes to analyze cyber-situational awareness of an energy facility in three stages. There are i) analysis of cyber threats to the energy infrastructure; ii) modeling of extreme situations scenarios in the energy sector caused by the implementation of the cyber threats; iii) risk assessment of the cybersecurity disruption to energy infrastructure. Three methods are presented, corresponding to each stage. The authors propose to apply semantic modeling methods to analyze the impact of cyber threats to energy facilities, taking into account energy security within the presented approach. Such methods show their effectiveness in the absence or incompleteness of data for modeling the behavior of systems, which defies formal description or accurate forecasting. The presented approach to the cyber situational awareness analysis of energy facilities considered as a synthesis of cybersecurity and situational awareness studies, characterized by the use of semantic modeling methods.

Keywords

cyber situational awareness, extreme situations in the energy sector, semantic modeling methods, cyber threats

Acknowledgements

This work was executed within the framework of project on state task MESI SB RAS FWEU-2021-0007 no. AAAA-A21-121012090007-7. The studying of separated aspects was supported by RFBR grants no. 19-07-00351, no. 20-010-00204, no. 19-57-04003

For citation

Gaskova D. A., Massel A. G. The Method of Cyber Awareness Analysis of an Energy Facility. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 17–28. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-17-28

Введение

Актуальность исследований киберситуационной осведомленности в энергетическом секторе обусловлена, с одной стороны, активно развивающейся тенденцией цифровой трансформации энергетики, а с другой – неполнотой информации об инцидентах, вызванных киберугрозами и разрозненностью руководящих документов в рассматриваемой области, что, в свою очередь, повышает риски внедрения новых информационных технологий в энергетике. Исследования развития энергетического сектора связаны с цифровой трансформацией энергетики, сопровождающейся разработкой и внедрением цифровых технологий, в состав которых включают промышленный Интернет вещей, 3D-моделирование, моделирование и прогнозирование на основе анализа «больших данных» (Big Data), нейросети, облачные и туманные вычисления, виртуальную и дополненную реальность, машинное обучение, компьютерную имитацию на основе цифровых двойников, интеллектуальные датчики, роботизацию производства, аддитивные технологии [1]. Перспективными технологиями в данной области также называют онтологические модели деятельности и распределенные реестры (Blockchain) [2] [1]. Современные решения для автоматизации технологического процесса на энергетических объектах становятся все более сложными и используют передовые цифровые технологии [2], что приводит к увеличению рисков нарушения безопасности этих объектов, вплоть до возникновения экстремальных ситуаций (ЭкС).

Активное развитие и внедрение интеллектуальных технологий на предприятиях послужило предпосылкой появления такого направления исследований, как киберситуационная осведомленность (КСО). Киберситуационная осведомленность (Cyber Situational Awareness) – область исследований, связанная с применением методов искусственного интеллекта в кибербезопасности, направленная на повышение осведомленности о возможных ситуациях нарушений кибербезопасности и автоматическое обнаружение киберугроз [3]. Под термином «киберситуационная осведомленность энергетических объектов» будем понимать осведомленность о состоянии киберсреды энергетических объектов, включающую информацию: о критических уязвимостях энергетических объектов с точки зрения кибербезопасности; о киберугрозах, инициирующих эти критические уязвимости, а также о техногенных угрозах энергетической безопасности, вызванных киберугрозами.

Методики анализа киберситуационной осведомленности об энергетических объектах

Анализ киберситуационной осведомленности об энергетических объектах в нотации IDEF0, отображающей структуру и функции анализа, а также потоки информации и материальных объектов, преобразуемые этими функциями, представлен на рис. 1. Для анализа киберситуационной

[1] Энергетическая Стратегия Российской Федерации на период до 2035 года: распоряжение Правительства Российской Федерации от 9 июня 2020 года № 1523-р // «Собрание законодательства РФ». 15.06.2020. № 24, ст. 3847.

[2] Концепция «Цифровая трансформация 2030» компании «РОССЕТИ», 2018 год. URL: https://www.rosseti.ru/investment/Kontseptsiya_Tsifrovaya_transformatsiya_2030.pdf.

Рис. 1. Контекстная диаграмма анализа киберситуационной осведомленности об энергетических объектах в нотации IDEF0

Fig. 1. Context diagram of the cyber situational awareness analysis of energy facilities in IDEF0 notation

Рис. 2. Декомпозиция первого уровня анализа киберситуационной осведомленности об энергетических объектах в нотации IDEF0

Fig. 2. First level decomposition diagram of the cyber situational awareness analysis of energy facilities in IDEF0 notation

осведомленности об энергетических объектах предложены методики, построены модели структурирования знаний, а именно фрактальная структурированная модель (ФС-модель) [4] и система онтологий [5], выделены источники информации и предложены инструментальные программные средства поддержки этого анализа.

Предлагаемые методики разработаны в соответствии со стандартом ИСО/МЭК 27005-2010 на основе методик анализа угроз и оценки рисков информационно-технологической безопасности энергетических комплексов [6] и моделирования угроз энергетической безопасности с помощью байесовских сетей доверия (БСД) [7].

Основные этапы анализа киберугроз и оценки рисков нарушения кибербезопасности энергетической инфраструктуры представлены на рис. 2. На первом уровне декомпозиции анализа киберситуационной осведомленности энергетических объектов выделены три основных процесса, представленные блоками на диаграмме. Анализ киберугроз энергетической инфраструктуры предлагается выполнять по соответствующей методике с использованием предложенной экспертной системы и дополнительных инструментальных средств, обычно применяемых при проведении аудита безопасности критической информационной инфраструктуры (КИИ). Моделирование сценариев ЭкС в энергетике, вызванных реализацией киберугроз, предлагается выполнять по соответствующей методике с использованием компонента БСД и инструментальных средств, реализующих математическое моделирование энергетических объектов. Оценку рисков предлагается осуществлять с учетом предыдущих процессов, используя компонент оценки рисков. Предлагаемые методики подробно рассмотрены далее.

Методика анализа киберугроз энергетической инфраструктуры включает три этапа.

1. *Описание энергетического объекта и определение критериев оценки* включают описание основных характеристик энергетического объекта (ЭО), бизнес-процессов и КИИ ЭО, в рамках которых определяются системы, их основные компоненты и функциональное назначение, в дальнейшем используемые при формировании предположений о возможных киберфизических последствиях реализации угроз [8]. Для каждого компонента системы определяется состав основных активов (аппаратно-программные, программные, протоколы и пр.). Предлагается в качестве основных критериев оценки угроз и уязвимостей определять критерий значимости (критичности) актива и критерий оценивания уязвимостей и угроз в соответствии со шкалами оценки. Результатом этапа является список активов и их качественная оценка по установленным критериям оценивания.

Для обеспечения приемлемого уровня КСО показатели безопасности должны быть приведены в соответствие с отраслевыми стандартами управления безопасностью компьютеров и сетей, а также с общими организационными и бизнес-целями в корпоративных средах [9] и технологических сегментах локальной вычислительной сети (ЛВС). В работе [9] выделяют 14 метрик безопасности корпоративной сети, где для каждой метрики определены шкала параметров и метод вычисления. В работе [10] представлен метод оценки рисков кибербезопасности на основе теории нечетких множеств, включающий семантические описания шкал выделенных факторов риска.

2. *Анализ уязвимостей и угроз КИИ объекта* включает анализ КИИ с последующим оцениванием активов КИИ, выявлением критически важных активов, выявлением киберуязвимостей в активах КИИ и киберугроз, выделение возможных целевых активов кибератаки. Результатом этапа является список уязвимостей для каждого значимого актива КИИ рассматриваемого объекта и киберугроз, которые могут эти уязвимости реализовать.

3. *Построение модели сценариев ЭкС в энергетике, вызванных киберугрозами*, в виде правил «ЕСЛИ-ТО», которые описывают возможные способы нарушения кибербезопасности энергетического объекта. Включает описание реализации киберугрозы, выделение векторов проникновения и векторов атак на целевые активы, а также их анализ. При этом векторы атак направлены на целевые активы, являющиеся составляющей технологической инфраструктуры энергетического объекта. Результатом этапа являются сценарии (вида цепочки правил

«ЕСЛИ-ТО») реализации всех уязвимостей, представленные цепочками уязвимостей и угроз, приводящих к нарушению нормального функционирования технологической инфраструктуры ЭО и некоторым негативным последствиям.

Методика моделирования сценариев экстремальных ситуаций в энергетике, вызванных реализацией киберугроз, включает четыре этапа.

1. *Формирование узлов и их взаимосвязей в графе БСД-модели* включает построение графа, состоящего из вершин нескольких типов, в соответствии с разработанной структурой сценария (рис. 3), и взаимосвязей между ними на основе построенных ранее правил вида «ЕСЛИ-ТО» (см. далее, формула (1)).

Структура сценария ЭкС в энергетике, вызванных реализацией киберугроз, разработана в соответствии с моделью [11], представлена на рис. 3 и включает следующие типы концептов:

- **уязвимости** (V) – множество всех обнаруженных критических уязвимостей рассматриваемой ЛВС;
- **киберугрозы** (T) – множество киберугроз активов ЛВС;
- **техногенные угрозы** (W) – множество техногенных угроз энергетической безопасности (ЭБ), вызванных киберугрозами T;
- **последствия** (C) – множество последствий реализации угроз T и W.

Структура сценария также включает классификацию концептов по типам, представленным выше, и по сегментам рассматриваемой ЛВС: гостевым, корпоративным, демилитаризованным зонам, технологическим сегментам.

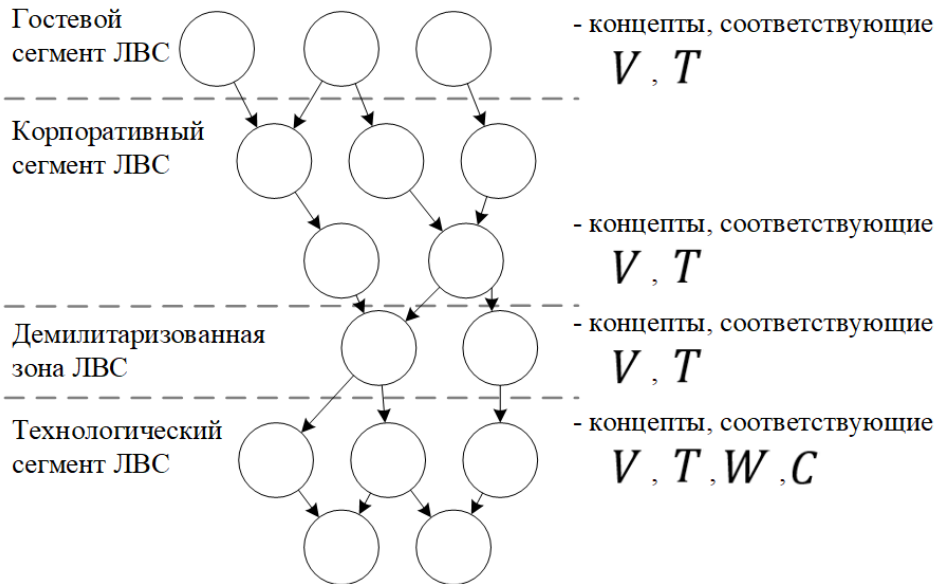


Рис. 3. Визуальное отображение структуры сценария ЭкС в энергетике, вызванных киберугрозами

Fig. 3. Visual display of the scenario structure of extreme situations in the energy sector caused by cyber threats

Такая структура способствует визуальному отображению возможных атак на технологический сегмент ЛВС из различных сегментов ЛВС. Например, в исследовании «Промышлен-

ные компании: векторы атак» [3] приводится типовая схема атаки на технологический сегмент, а в [12] описана модель атаки по этапам.

Графом БСД-модели будем называть ациклический, ориентированный, взвешенный граф G такой, что

$$G = (N, U; q), \tag{1}$$

где N – множество вершин графа, U – множество дуг графа, q – функция на вершинах.

Функция q на вершинах графа G задается как

$$q: N \to B, \tag{2}$$

где B_i – матрица весов вершины N_i. Для каждой вершины N_i элементами этой матрицы являются вероятности соответствующей случайной величины X_i.

2. *Определение вероятностных характеристик сценария* включает заполнение матриц весов для каждой вершины N_i (таблиц условных вероятностей (ТУВ)), на основе опыта эксперта или имеющейся накопленной информации о подобных инцидентах и формирование графа.

Каждой вершине N_i графа G соответствует единственная случайная величина X_i, такая, что

$$X_i \in \{X^V \cup X^T \cup X^W \cup X^C\}, \tag{3}$$

где $i = \overline{1,n}$ и $n = |V| + |T| + |W| + |C|$, X^V – множество случайных величин, соответствующих V, X^T – множество случайных величин, соответствующих T, X^W – множество случайных величин, соответствующих W, X^C – множество случайных величин, соответствующих C. Безусловные вероятности случайной величины X_i задаются в ТУВ в случае, когда у соответствующей вершины N_i предков нет (табл. 1). В случае, когда есть набор предков $pa(X_i)$ вершины N_i, задаются условные вероятности (табл. 2).

Формирование графа G сопровождается заполнением матриц весов B_i для вершин N_i, в данном случае $i = \overline{1,|N|}$. При этом совместное распределение вероятностей на множестве вершин, соответствующих X графа G, называют

$$\begin{gathered} P(X) = (P_1(X), P_2(X), \dots, P_n(X)), \\ P_i(X) = P(X_i | pa(X_i)), \end{gathered} \tag{4}$$

где X_i введено ранее в (3), $pa(X_i)$ – множество предков вершины, соответствующей X_i, в данном случае $n = |N|$.

Таблица 1

Общий вид ТУВ для отображения безусловных вероятностей случайной величины X_i

Table 1

General view of the table of conditional probabilities for displaying the unconditional probabilities of a random variable

X_i.	$\overline{X_i}$
T	F
p	$1-p$

[3] Промышленные компании: векторы атак / Positive Technologies, 2018. URL: https://www.ptsecurity.com/ru-ru/research/analytics/ics-attacks-2018.

Таблица 2

Общий вид ТУВ для отображения условных вероятностей случайной величины X_i

Table 2

General view of the table of conditional probabilities for displaying the conditional probabilities of a random variable

$pa(X_i)_1$	...	$pa(X_i)_k$	$X_i.$	$\overline{X_i}$
T	T	T	p_1	$1-p_1$
...	...	...	...	...
T	T	T	...	...
F	F	F	...	...
...	...	...	...	...
F	F	F	p_{2^k}	$1-p_{2^k}$

В таблице приняты следующие обозначения: p – вероятность случайной величины X_i, $k = |pa(X_i)|$, T (true) – состояние реализации уязвимости, угрозы или последствия, F (false) – противоположное T состояние.

3. *Проведение вероятностного эксперимента* включает вычисление распределения вероятностей в БСД в зависимости от наблюдаемых уязвимостей и угроз в построенных векторах атак. Проведение расчетов выполняется с целью ответа на вопрос «Как изменится вероятность последствий, если принять, что некоторое множество уязвимостей и / или угроз реализовано?». Такое множество будем называть множеством свидетельств, каждое из которых можно описать утверждением вида «Определенные уязвимости из V и / или угрозы из T, W реализованы». При наличии такого множества свидетельств необходимо пересчитать вероятности дискретных случайных величин X, соответствующих вершинам N графа G.

Вершины, в которые введены свидетельства, имеют множество вершин-потомков. Для того чтобы в каждой из таких вершин-потомков пересчитать апостериорную вероятность (после наблюдения), в их матрицах весов B требуется не учитывать все несовместимые со свидетельством в вершине-предке случаи. Если случай становится невозможным, то его вероятность принимается равной нулю (невозможное событие). Вероятности прочих случаев пересчитываются (блок 2) по формуле Байеса:

$$P(X_i|e) = \frac{\sum_{j=1}^{m} P_j(X,e)}{P(e)}, \quad (5)$$

где $P(X_i|e)$ – апостериорная вероятность X_i, E – множество дискретных случайных величин, являющихся свидетельствами, e – значения этих случайных величин, m – количество оставшихся случайных переменных, в которые не введены свидетельства.

4. *Анализ альтернативных сценариев* включает анализ наиболее уязвимых мест, критических угроз и векторов атак.

Методика оценки рисков нарушения кибербезопасности энергетической инфраструктуры включает четыре основных этапа.

1. *Описание рисков* включает списки наблюдаемых уязвимостей и угроз, а также вероятность последствия:

$$R = \{T, V, W, C, D\}, \quad (6)$$

где значения T, V, W, C введены ранее, D – множество ущербов последствий сценариев.

2. *Оценивание рисков* включает расчет и дальнейшее качественное и количественное оценивание. Количественную оценку рисков предлагается выполнять в отношении последствия сценария S_i по формуле

$$R_i = P(c_i) \times D_i, \quad (7)$$

где $P(c_i)$ – вероятность последствия сценария S_i, рассчитываемая по формуле (5), D_i – ущерб от реализации киберугроз этого сценария, $i = \overline{1, 2^n - 1}$.

Принимается, что существует взаимно-однозначное соответствие между сценарием S_i и вероятностью его последствия $P(c_i), i = \overline{1, 2^n - 1}, n = |V| + |T| + |W|$. Условия оценки сценариев ЭкС, вызванных киберугрозами, имеют вид, представленный в табл. 3.

Таблица 3

Условия оценки сценариев ЭкС в энергетике, вызванных киберугрозами

Table 3

Conditions for assessing extreme situations in energy sector scenarios caused by cyber threats

Незначительный уровень опасности	Средний уровень опасности	Высокий уровень опасности
$\begin{cases} 0 \le P(\mathrm{c}_i) \times D_i < l_1 \\ 0 \le P(\mathrm{c}_i) \le 1 \\ 0 \le D_i \le D^{\mathrm{m}} \end{cases}$ (8)	$\begin{cases} l_1 \le P(\mathrm{c}_i) \times D_i < l_2 \\ 0 \le P(\mathrm{c}_i) \le 1 \\ 0 \le D_i \le D^{\mathrm{m}} \end{cases}$ (9)	$\begin{cases} l_2 \le P(\mathrm{c}_i) \times D_i \\ 0 \le P(\mathrm{c}_i) \le 1 \\ 0 \le D_i \le D^{\mathrm{m}} \end{cases}$ (10)

В таблице приняты следующие обозначения: l_1, l_2 – критерии кластеризации сценариев ЭкС, вызванных киберугрозами, D^{m} – максимальный ущерб, $i = \overline{1, 2^n - 1}$.

Все сценарии разбиваются на три кластера: норма, предкризис, кризис:

$$K_s = \{S_{s1}, S_{s2}, \dots, S_{sl}\}, \quad (11)$$

где K_s – множество кластеризованных сценариев ЭкС в энергетике, вызванных киберугрозами, $s = \overline{1,3}$.

3. *Ранжирование активов КИИ и выработка рекомендаций* включает составление списка активов по критерию значимости, по уровню рисков и выработку соответствующих мер по их снижению.

4. *Определение уровня киберситуационной осведомленности* энергетического объекта включает подсчет отношения количества рассмотренных сценариев к количеству возможных сценариев в модели по формуле

$$L = \frac{\gamma}{2^n - 1}, \quad (12)$$

где L – уровень КСО энергетического объекта, γ – количество рассчитанных сценариев, $2^n - 1$ – общее количество сценариев, $n = |V| + |T| + |W|$.

Заключение

В статье предложена методика анализа киберситуационной осведомленности об энергетическом объекте, основанная на вероятностной модели сценариев ЭкС в энергетике, вызванных реализацией киберугроз. Модель, построенная на основе аппарата байесовских сетей доверия, включает в себя структуру сценариев. Описана структура сценария: типы концептов и их распределение по сегментам ЛВС. Предложена метрика оценки КСО, основанная на формировании векторов атак на рассматриваемую ЛВС и байесовском рассуждении на сети.

Список литературы

1. **Zhang C.**, **Romagnoli A.**, **Zhou L.**, **Kraft M.** From Numerical Model to Computational Intelligence: The Digital Transition of Urban Energy System. *Energy Procedia*, 2017, vol. 143, p. 884–890. DOI 10.1016/j.egypro.2017.12.778
2. **Irmak E.**, **Erkek I.** An overview of cyber-attack vectors on SCADA systems. In: 6th International Symposium on Digital Forensic and Security (ISDFS). March, 2018. DOI 10.1109/isdfs.2018.8355379
3. **Frank U.**, **Brynielsson J.** Cyber Situational Awareness – A systematic review of literature. *Computer Security*, 2014, vol. 46, p. 18–31. DOI 10.1016/j.cose.2014.06.008
4. **Gaskova D.** Fractal Stratified Model Development for Critical Infrastructure from the standpoint of Energy and Cyber Security. In: Proceedings of the VI International Workshop "Critical Infrastructures: Contingency Management, Intelligent, Agent-Based, Cloud Computing and Cyber Security (IWCI 2019)". Irkutsk, Atlantis Press, 2019, p. 179–183. DOI 10.2991/iwci-19.2019.31
5. **Гаськова Д. А.**, **Массель А. Г.** Онтологический инжиниринг анализа угроз кибербезопасности критических инфраструктур // Онтология проектирования. 2019. Т. 9, № 2 (32). С. 225–238. DOI 10.18287/2223-9537-2019-9-2-225-238
6. **Массель Л.В.**, **Пяткова Е.В.** Применение байесовских сетей доверия для интеллектуальной поддержки исследований проблем энергетической безопасности // Вестник ИрГТУ. 2012. № 2. С. 8–13.
7. **Массель А. Г.** Методика анализа угроз и оценки риска нарушения информационно-технологической безопасности энергетических комплексов // Тр. XX Байкальской Всерос. конф. Иркутск: ИСЭМ СО РАН, 2015. Т. 3. С. 186–195.
8. **Дащенко Ю**. Моделирование угроз в условиях методической неопределенности / Kaspersky Lab ICS CERT. URL: https://ics-cert.kaspersky.ru/media/KL-ICS-CERT-Model-ugroz.pdf.
9. **Cheng Y.**, **Deng J.**, **Li J.**, **DeLoach S. A.**, **Singhal A.**, **Ou X.** Metrics of Security. In: Kott A., Wang C., Erbacher R. (eds.). Cyber Defense and Situational Awareness. Advances in Information Security, 2014, vol. 62. Springer, Cham. DOI 10.1007/978-3-319-11391-3_13
10. **Колосок И. Н.**, **Гурина Л. А.** Оценка рисков кибербезопасности информационно-коммуникационной инфраструктуры интеллектуальной энергетической системы // Информационные и математические технологии в науке и управлении. 2019. № 2 (14). С. 40–51. DOI 10.25729/2413-0133-2019-2-04
11. **Гаськова Д. А.** Метод определения уровня киберситуационной осведомленности энергетических объектов // Информационные и математические технологии в науке и управлении. 2020. № 4 (20). С. 64–74. DOI 10.38028/ESI.2020.20.4.006
12. **Assante M. J.**, **Lee R. M.** The Industrial Control System Cyber Kill Chain. URL: https://www.sans.org/reading-room/whitepapers/ICS/industrial-control-systemcyber-kill-chain-36297.

References

1. **Zhang C.**, **Romagnoli A.**, **Zhou L.**, **Kraft M.** From Numerical Model to Computational Intelligence: The Digital Transition of Urban Energy System. *Energy Procedia*, 2017, vol. 143, p. 884–890. DOI 10.1016/j.egypro.2017.12.778
2. **Irmak E.**, **Erkek I.** An overview of cyber-attack vectors on SCADA systems. In: 6th International Symposium on Digital Forensic and Security (ISDFS). March, 2018. DOI 10.1109/isdfs.2018.8355379
3. **Frank U.**, **Brynielsson J.** Cyber Situational Awareness – A systematic review of literature. *Computer Security*, 2014, vol. 46, p. 18–31. DOI 10.1016/j.cose.2014.06.008

4. **Gaskova D.** Fractal Stratified Model Development for Critical Infrastructure from the standpoint of Energy and Cyber Security. In: Proceedings of the VI International Workshop “Critical Infrastructures: Contingency Management, Intelligent, Agent-Based, Cloud Computing and Cyber Security (IWCI 2019)”. Irkutsk, Atlantis Press, 2019, p. 179–183. DOI 10.2991/iwci-19.2019.31
5. **Gaskova D. A.**, **Massel A. G.** Ontological engineering for the development of the intelligent system for threats analysis and risk assessment of cybersecurity in energy facilities. *Ontology of designing*, 2019, vol. 9 (2), p. 225–238. (in Russ.) DOI 10.18287/2223-9537-2019-9-2-225-238
6. **Massel L. V.**, **Pyatkova E. V.** Application of Bayesian Networks to Intelligently Support Energy Security Research. *Proceedings of Irkutsk State Technical University*, 2012, no. 2, p. 8–13. (in Russ.)
7. **Massel A. G.** Methods of the analysis of threats, risk assessment violations of information and technological security of energy complexes. In: Proceedings of the XX Baikal All-Russian Conference “Information and mathematical technologies in science and management”. Irkutsk, MESI SB RAS, 2015, vol. 3, p. 186–195. (in Russ.)
8. **Dashchenko Yu.** Threat modeling in conditions of methodological uncertainty. Kaspersky Lab ICS CERT. URL: https://ics-cert.kaspersky.ru/media/KL-ICS-CERT-Model-ugroz.pdf (in Russ.)
9. **Cheng Y.**, **Deng J.**, **Li J.**, **DeLoach S. A.**, **Singhal A.**, **Ou X.** Metrics of Security. In: Kott A., Wang C., Erbacher R. (eds.). Cyber Defense and Situational Awareness. Advances in Information Security, 2014, vol. 62. Springer, Cham. DOI 10.1007/978-3-319-11391-3_13
10. **Kolosok I. N.**, **Gurina L. A.** Cybersecurity Risk Assessment of Information and Communication Infrastructure of Intelligent Energy System. *Information and mathematical technologies in science and management*, 2019, no. 2 (14), p. 40–51. (in Russ.) DOI 10.25729/2413-0133-2019-2-04
11. **Gaskova D. A.** Method for Determining the Level of Cyber Situational Awarenes on Energy Facilities. *Information and mathematical technologies in science and management*, 2020, no. 4 (20), p. 64–74. (in Russ.) DOI 10.38028/ESI.2020.20.4.006
12. **Assante M. J.**, **Lee R. M.** The Industrial Control System Cyber Kill Chain. URL: https://www.sans.org/reading-room/whitepapers/ICS/industrial-control-systemcyber-kill-chain-36297.

Материал поступил в редколлегию
Received
11.02.2021

Сведения об авторах

Гаськова Дарья Александровна, младший научный сотрудник отдела систем искусственного интеллекта в энергетике Федерального государственного бюджетного учреждения науки Институт систем энергетики им. Л. А. Мелентьева Сибирского отделения Российской академии наук (Иркутск, Россия)

gaskovada@gmail.com

Массель Алексей Геннадьевич, кандидат технических наук, старший научный сотрудник отдела систем искусственного интеллекта в энергетике Федерального государственного бюджетного учреждения науки Институт систем энергетики им. Л. А. Мелентьева Сибирского отделения Российской академии наук (Иркутск, Россия)

amassel@gmail.com

Information about the Authors

Daria A. Gaskova, Junior Fellow of Department of Artificial Intelligence Systems in the Energy Sector, Melentiev Energy Systems Institute Siberian Branch of the Russian Academy of Sciences (Irkutsk, Russian Federation)

gaskovada@gmail.com

Aleksei G. Massel, PhD in Engineering Science, Senior researcher of Department of Artificial Intelligence Systems in the Energy Sector, Melentiev Energy Systems Institute Siberian Branch of the Russian Academy of Sciences (Irkutsk, Russian Federation)

amassel@gmail.com

УДК 004.43
DOI 10.25205/1818-7900-2021-19-2-29-52

Визуализация результатов анализа языков программирования для их поверхностного сравнения

Л. В. Городняя

Новосибирский государственный университет
Новосибирск, Россия

Институт систем информатики им. А. П. Ершова СО РАН
Новосибирск, Россия

Аннотация

Статья посвящена выбору наглядной и лаконичной формы для представления результатов анаиза и сравнения языков программирования, удобной для оценки выразительной силы языков и трудоемкости реализации систем программирования. Формализация приспособлена к парадигмальному анализу определений языков программирования и выбору практичных критериев декомпозиции программ. В качестве основного подхода выбрана семантическая декомпозиция определений языков в рамках анализа парадигм программирования. Такой выбор позволяет выделять автономно развиваемые типовые компоненты программ, которые могут быть приспособлены к конструированию различных информационных систем. Многие работы по методам разработки программных систем зависят от практичности подходов к декомпозиции программ, отлаживаемых с помощью систем программирования. Решение этой проблемы полезно при изучении методов программирования, исследовании истории языков программирования, для сравнения парадигм программирования, потенциала используемых схем и моделей, оценки уровня новизны создаваемых языков программирования, а также при выборе критериев декомпозиции программ. Кроме того, определенность критериев позволяет формировать методику обучения разработке компонентов информационных систем. Попутно показана дистанция в понятийной сложности между программированием и разработкой систем программирования.

Ключевые слова

определение языков программирования, парадигмы программирования, декомпозиция программ, критерии декомпозиции, семантические системы

Благодарности

Работа выполнена при поддержке РФФИ, проект № 18-07-01048-а

Для цитирования

Городняя Л. В. Визуализация результатов анализа языков программирования для их поверхностного сравнения // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 29–52. DOI 10.25205/1818-7900-2021-19-2-29-52

Visualization of the Results of the Analysis of Programming Languages for Their Superficial Comparison

L. V. Gorodnyaya

Novosibirsk State University
Novosibirsk, Russian Federation

A. P. Ershov Institute of Informatics Systems SB RAS
Novosibirsk, Russian Federation

Abstract

The article is devoted to the choice of a clear and concise form for presenting the results of analise and comparing programming languages and systems, convenient for assessing the expressive power of languages and the complexity of implementing systems. The formalization is adapted to the paradigm analysis of the definitions of programming languages and the selection of practical criteria for decomposition of programs. The semantic decomposition of the definitions of languages and systems as part of the analysis of programming paradigms was chosen as the main ap-

proach. Such a choice makes it possible to single out autonomously developed typical program components that can be adapted to the design of various information systems. Many works on methods for developing software systems depend on the practicality of approaches to decomposition of programs debugged using programming systems. The solution to this problem is useful when studying programming methods, studying the history of programming languages, for comparing programming paradigms, the potential of used circuits and models, assessing the novelty level of created programming languages, and also when choosing criteria for program decomposition. In addition, their existence allows us to form a teaching methodology for developing the components of information systems. Along the way, the distance in conceptual complexity between programming and programming system development is shown.

Keywords

definition of programming languages, programming paradigms, program decomposition, decomposition criteria, semantic systems

Acknowledgements

The study was funded by RFBR according to the research project no. 18-07-01048-a

For citation

xxx. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 29–52. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-29-52

Введение

В данной статье предлагается ряд решений для визуального представления и формализованного измерения понятийной сложности конструкций, поддержанных в определениях языков и систем программирования (ЯиСП). Такое представление может быть применено при сравнении парадигм программирования (ПП), потенциала схем и моделей, используемых при разработке программ, оценки уровня новизны создаваемых языков программирования (ЯП), а также при выборе критериев декомпозиции программ. Долгоживущие программы, такие как системы программирования (СП), требуют объективных критериев декомпозиции программ, отражающих возможность автономного развития выделенных компонентов СП с целью профилактики повторного программирования при развитии реализуемого ЯП.

1. Математические абстракции

В качестве отправной точки для представления решений по декомпозиции программ можно принять классическое понятие «алгебраическая система» [1]: алгебраическая система – это <A, F>, где A – основное множество значений, F – конечный набор операций над множеством A.

Примерно так можно формализовать библиотечные модули в СП и классы объектов в ООП до тех пор, пока при реализации ЯП не возникает необходимость использовать разные модели вычислений по одним и тем же формулам. С. С. Лавров предложил понятие «семантическая система», расширяющее понятие «алгебраическая система» заданием явного правила применения функций к значениям [2]: семантическая система – это <**V**, **F**, **R**>, где **V** – основное множество значений, **F** – конечный набор функций, возможно принадлежащих основному множеству, **R** – правило применения функций к значениям, возможно входящее в набор функций.

Близкое понятие под названием «монады» введено в язык Haskell.

Большинство ЯП поддерживает разные варианты утончения понятия «семантическая система», например, где **F** – конечный набор функций, **не принадлежащих** основному множеству, **R** – правило применения функций к значениям, **не входящее** в набор функций.

Соответствующее обозначение такого вида семантических систем можно представить как <**V; F; R**>. Вид <V; F; R> характерен для ЯП, обладающих четким разделением данных и программ, а также для языков управления базами данных.

2. Архитектурные модели

Следует отметить, что жаргон современного практического программирования понятие «язык программирования» использует как «входной язык типовой системы программирования». Разница заключается в том, что СП обычно сопровождает реализацию ЯП расширяемым комплектом библиотечных модулей. В результате происходит сглаживание видимых различий между языками программирования и системами программирования. Кроме того, именно доступная СП является истиным документом по языку программирования, в отличие от формальных описания и станддартов, слабо представляющих реализационную прагматику ЯП и часто по объему превышающих разумные размеры. Это приводит к целесообразности при анализе особенностей ЯП и структурировании поддержанных в нем средств учитывать архитектурные модели.

Успехи эелементной базы не исключают актуальность создания эффективных программ. Поэтому выбор параметров декомпозиции программ должен учитывать особенности архитектурных моделей. Наиболее известные компьютерные архитектуры обычно представляют как конструкцию из вычислительного устройства (E), памяти (M), устройства управления процессами (C) и средств коммуникации между устройствами и их элементами (S). Такая конструкция допускает детализациею на уровне более тонких аппаратных решений и расширение на уровне периферийных устройств, что дает первое приближение для выбора основных видов семантических систем в языках программирования [3]. Рассматривая любые семантические системы, важно отметить формальную разницу в характере выполнения функций таких систем. Так, для любого множества значений V реализационно различимы виды функций для методов вычислений FE: (V* → V+), средств доступа к памяти FM: (T : N → V), особенностей управления вычислениями FC: (F → {0, 1})* и обратимой комплексации и структурирования данных FS: (A → K) U (A ← K) = (A ↔ K).

Следует обратить внимание, что такие виды семантических систем часто обладают кумулятивным эффектом в порядке «VEMCS». Если V – произвольное множество значений, то FE: (V* → V+) – обычные вычисления, заданные формулами над значениями из этого множества, а формула может представлять самоопределимое константное значение из V. Для реализации средств доступа к памяти FM: (T : N → V) характерно выделение понятия «адрес» (N) и подразумевается существование специальной таблицы T, по которой определено соответствие адресов заданным значениям, при задании которых могут использоваться формулы вычислений. Особенности управления вычислениями FC: (F → {0,1})* состоят в разметке запрограммированных действий для выделения выполняемых, обычно используя понятие адресуемой памяти. Обратимая комплексация или структурирование данных в современных архитектурах FS: (A ↔ K) требует определения границы между атомарными (A) и сложными, конструируемыми (K) объектами с возможностью как наращивания сложности, так и урощения любых построений с учетом разных условий.

Это приводит к представлению о кумулятивной шкале кактегорий семантических систем на основе классификации видов функций (табл. 1).

Таблица 1

Классы семантических систем

Table 1

Classes of semantic systems

№ столбца	V	E	M	C	S
Виды функций (F V)	**V**	**FE: V* → V+**	**FM: (T : N → V)** V = N U V	**FC: (F → {0, 1})*** [1] V = {0,1} U V	**FS: (A → K) U (A ← K)** V = A U K

[1] Обозначение уточнено А. В. Климовым.

3. Понятийная матрица

Одним и тем же наборам функций могут соответствовать разные правила **R**, определяющие методы вычислений, влияющие на результативность выполнения программ.

RE – обычные арифметические вычисления, отображающие произвольный ряд **значений** аргументов в не менее чем один результат; механизм вычисления можно не задавать, считать, что происходит чудо выдачи результата средствами аппаратуры или подобно им.

RM – символьные вычисления, подставляющие **представления** аргументов без их предварительного вычисления; механизм сводится к разным схемам копирования данных, обычно доступным в любой архитектуре.

RC – частичные или смешанные вычисления, лавирующие между вычислением и подстановкой в зависимости от разных условий; обычно в числе таких условий имеются прерывания, имеющие аппаратную поддержку.

RS – обобщенные и / или параллельные вычисления, оперирующие организацией процессов как множеством потоков над комплексами из разных устройств или данных; обычно поддерживаются особо эффективные механизмы доступа к соседям.

Представление таких различий можно выразить, дополнив горизонтальную шкалу видов функций вертикальной шкалой моделей вычислений или методов применения функций к значениям (R), что будет выглядеть как матрица семантических систем – понятийная матрица. Можно вспомнить, что ЯП определяется не только на уровне лексики, синтаксиса и семантики, но еще и на уровне **прагматики**. Исторически прагматика представлена как традиция реализации ЯП в наиболее популярных СП в форме правил, уточныющих особенности категорий семантических систем, поддерживающих разные парадигмы программирования. Следует отметить, что разные парадигмы могут быть реализованы одной и той же прагматикой. Это делает нетривиальным проблему представления уловия для различия парадигм [4]. Различия категорий семантических систем, поддерживающих различимые парадигмы на уровне правил применения функций к значениям в наиболее известных языках высокого уровня (ЯВУ), выражены табл. 2–5. Строка «RE: ядро» этой матрицы представляет уровень базовой семантики ЯП, существенные различия видов функций семантических систем которого можно представить табл. 2. **Базовая семантика** – это подмножество определения ЯП, из которого можно вывести все остальные его средства методом консервативного расширения, т. е. введения функций над базовой семантикой.

Таблица 2

Семантическая декомпозиция минимального ядра ЯП

Table 2

Semantic decomposition of the minimal PL kernel

№ столбца	V	E	M	C	S
Виды функций	V	FE: V* → V+	FM: (T : A → V)	FC: {F → {0,1}}	FS: A ↔ K
RE: Ядро	Значение	Операции	Память	Управление	Вектор

Ядро – семантический базис. Полное определение ЯП можно получать как консервативное расширение ядра. Обычно ядро приспособлено и к неконсервативному расширению пополнением набора библиотечных функций, реализуемых на уровне аппаратуры. Это позволяет в реализации СП для любого ЯП поддерживать разные парадигмы программирования, необходимые для обеспечения полного жизненного цикла программ, чтобы достигать результата независимо от исходных возможностей ЯП.

Значение – минимальное представление объектов из области приложения языка, обычно это самоопределимые константы.

Операции – минимальный комплект функций для обработки значений.

Память – введение адресов для лаконичного и уникального представления значений (указатели, идентификаторы, переменные, метки).

Управление – разметка выполнимости композиции элементов программы из (операций, функций, действий и т. п.) специально выбранными значениями, например, {0|1} или {True | False}, или {Nil | T}.

Вектор – обратимое конструирование одноуровневых комплектов, рассматриваемых как целостность, из которых можно восстанавливать исходные элементы. На уровне ядра достаточно одной структуры, поддерживающей доступ по принципу соседства, – вектора, списка, очереди или т. п.

Строка «RM: Макро» табл. 3 представляет уровень макрорасширений, что вместе с первой строкой достаточно, чтобы характеризовать основные средства большинства языков низкого уровня.

Таблица 3

Семантическая декомпозиция макрорасширения ядра ЯП

Table 3

Semantic decomposition of the macro-extension of the PL core

№ столбца	V	E	M	C	S
Виды функций	V	FE: V* → V+	FM: (T : A → V)	FC: {F → {0,1}}	FS: A ↔ K
RE: Ядро	Значение	Операции	Память	Управление	Вектор
RM: Макро	Данное	Функции	Задание	Блоки	Стек

Макро – пополнение ядра средствами обработки представлений, используемых с целью укрупнения любых конструкций, что позволяет выполнять консервативное расширение ЯП. Кроме того, оно способствует лаконизму текстов программ. Макротехника позволяет наследовать отлаженность фрагментов программ. Бывает важным исключать дубли частей текста, кода и структур данных. Простейший механизм макрогенерации обычно присутствует во многих СП как препроцессор. Так же бывает устроена техника кодогенерации и обработки шаблонов при компиляции программ, нередко проникающая и на уровень ЯП. Реализация укрупнений может быть функционально эквивалентна вызову подпрограмм. Взаимозаменяемость макроподстановки и вызова подпрограмм нередко используется при оптимизации программ.

Данное – хранимое значение или выражение, допускающее уникальность экземпляра, доступного многократно по адресу, возможно, на внешнем устройстве.

Функции – укрупнение операций с возможной параметризацией операндов. Реализационная прагматика может отличаться техникой передачи параметров через стек или специальное поле аргументов или неявно. Последнее позволяет и работу с памятью формально рассматривать как функцию с неявным аргументом, выполняющим роль функции T, задающей соответствия адресов и значений.

Задание – хранимое именованное выражение с возможностью многократного выполнения.

Блоки – хранимое выражение или код программы, представляющий составные действия, ветвления, циклы, вызовы функций, обычно с локализацией переменных, приводящей к понятию «иерархия».

Стек – схема организации данных, дополняющей соседство определенной дисциплиной доступа для поддержки иерархии вычислений, возможно, с защитой независимых блоков.

Повышение уровня ЯП обеспечивается не только особым вниманием к средствам укрупнения данных на базе понятия «иерархия» и оперирование блоками программы. Дальнейшее наращивание объемов разрабатываемых программ отчасти достигается автоматизацией контроля некоторых условий корректности применения операций и функций к их операндам в определенных границах. Становятся важными понятия «предикат» и «тип переменных», удобно проверяемые при компиляции, что представлено строкой «RC: Границы» табл. 4. Разновидностью условий являются сигналы о прерываниях или вычисляемых неподходящих ситуациях.

Таблица 4

Семантическая декомпозиция диагностического дополнения ядра ЯП

Table 4

Semantic decomposition of the diagnostic complement of the PL kernel

№ столбца	V	E	M	C	S
Виды функций	V	FE: V* → V+	FM: (T : A → V)	FC: {F → {0,1}}	FS: A ↔ K
RE: Ядро	Значение	Операции	Память	Управление	Вектор
RM: Макро	Данное	Функции	Задание	Блоки	Стек
RC: Границы	Исключения	Предикаты	Типы переменных	Логика	Варианты

Границы – методы проверки вычислимости функций и выполнимости заданий. Цель представления границ – снижение трудоемкости отладки программ упрощением поиска ошибок. При отсутствии ошибок проверка воспринимается как накладные расходы. Встречаются механизмы установки ловушек на непредусмотренные ситуации и программирования обработки исключений с возможностью продолжения вычислений.

Исключения – выбор специальных значений для разметки неожиданных ситуаций. В некоторых ЯП вводят значения типа Error. При переходе к СП происходит добавление текстовых шаблонов для формирования диагностических сообщений об исключительных ситуациях.

Предикаты – специальные функции, позволяющие определять типы значений или сравнивать значения независимо от расположения данных в памяти. Роль предиката может выполнять любая функция при подходящих договоренностях и схеме реагирования на ее результаты.

Типы переменных – связывание типа значения с переменной, хранящей нетипизированный код значения в памяти.

Логика – проверка соответствия типа данных или значений операциям обработки значений или доступа к памяти, возможно, с учетом условий вычислимости.

Варианты – схема организации данных без определенной дисциплины доступа для организации перебора или выбора равноправных элементов или блоков. Полезно при отладке программ как механизм удостоверения принципиальной выполнимости вычислений при частичной постановке задачи.

По мере расширения границ программа может стать достаточно универсальной, способной к разумному поведению на любых входных данных, что выражено строкой «RS: Общность» в табл. 5, в которой добавлена слева колонка с обозначениями строк полученной понятийной матрицы.

Таблица 5

Понятийная матрица:
семантическая декомпозиция практичного обобщения ядра ЯП

Table 5

Concept matrix:
Semantic decomposition of a practical generalization of the PL core

№ строки	№ столбца	V	E	M	C	S
	Виды функций	V	FE: V* → V+	FM: (T : A → V)	FC: {F → {0,1}}	FS: A ↔ K
E	RE: Ядро	Значение	Операции	Память	Управление	Вектор
M	RM: Макро	Данное	Функции	Задание	Блоки	Стек
C	RC: Границы	Исключения	Предикаты	Типы переменных	Логика	Варианты
S	RS: Общность	Неопределенность	Мульти-операции	Внешний мир	Отображения	Ввод-вывод

Общность – дополнительные средства обеспечения отладки и применения программ, поддерживающие возможность разумного продолжения вычислений при любых исходных данных и аварийных ситуациях.

Неопределенность – вводятся специальные дополнительные значения и ловушки (_|_, Error, Future). Такое расширение множества значений позволяет учитывать в текстах программ некоторые отдельные особенности процесса разработки, отладки и схемы жизненного цикла программ.

Мультиоперации – допускается произвольное число операндов операций, аргументов и результатов функций. Возможно просачивание определений на однородные структуры данных, позволяющее определения над элементарными данными автоматически распространять на более сложные данные.

Внешний мир – механизм неявного расширения области действия операций и функций на периферийные устройства, рассматриваемые как обобщение памяти.

Отображение – возможность регулярного применения функции к серии данных благодаря использованию представлений функций или указателей в качестве аргументов функций более высокого порядка.

Ввод-вывод – средства приема данных с внешних устройств и размещения данных на внешних носителях данных, включая средства доступа к устройствам с уровня программы. Стандартно имеется в виду прием данных с клавиатуры и изображение данных на экране. Обычно подразумевается аксиоматика, требующая совместимости форматов ввода-вывода: для всякого вводимого данного существует эквивалентное ему выводимое данное и, обратно, если данное может быть выведено, то его можно ввести без потерь.

Таким образом, разным видам функций относительно схемы применения функций к аргументам на уровне ЯП при реализации СП соответствуют определенные позиции специальной понятийной матрицы, полученной как двумерная кумулятивная шкала. Кумулятивный эффект по второму измерению получается совмещением результатов ячейки [E,S] с [M,V], [M,S] с [C,V], [C,S] с [S,V]. Более подробные пояснения даны в табл. 6. Понятийную матрицу можно равссматривать как форму онтологического определения ЯП.

Таблица 6

Кумулятивные эффекты в понятийной матрице при переходе на очередную позицию (наследование средств предшествующих позиций)

Table 6

Cumulative effects in the concept matrix when moving to the next position (inheritance of the means of previous positions)

Индекс позиции	Пояснение
EV	Самоопределяемые скаляры, их смысл не требует интерпретации, текстовое представление понятно без комментариев
EE	Операции над скалярами, они должны вырабатывать скаляры. При выходе за границы V могут его пополнять или вырабатывать сигнал неуспеха. Множество значений может быть пополнено представлениями формул
EM	Обработка памяти над таблицей адресов с соответствующими им значениями. Адреса и таблица могут быть значениями или не рассматриваться как значения и использоваться неявно как сущности другой природы. Появляется именование значений и формул, что расширяет класс допустимых формул, обеспечивает многократное использование хранимых результатов и приводит к понятию «данное»
EC	Безусловное и условное (без «else») управление процессом вычислений, приоритеты в формулах вычисления и переход к очередному действию или по метке
ES	Конструирование последовательностей по принципу соседства и перебора слева направо дополняются возможностью возвратов или выбора любого элемента ради обратимости. Как правило, это векторы или строки
MV	Хранимые данные могут иметь имена, что позволяет решать проблемы укрупнения используемых единиц, воспринимаемых равноправно со скалярами и операциями. Возникают имена логических значений для представления условий выбора хода процесса
ME	Композиции операций рассматриваются как безымянные функции, равноправные базовым операциям, они могут обрабатывать и вырабатывать не только скаляры, но любые определенные данные, используя упаковку ряда значений в последовательность
MM	Именование функций упрощает их многократное использование в формулах, включая представление рекурсии
MC	Композиции операций и функций можно рассматривать как одно действие, что приводит к понятию «блок», выделенный скобками, влияющими на порядок вычислений и задающими области видимости имен, – иерархия. Появляются системные процедуры, что может приводить к неконсервативным расширениям
MS	Структуры однородных данных и процессов сопровождаются дисциплиной доступа к элементам – FIFO, FILO, взаимоисключение, одновременность или др.
CV	Множество значений пополняется специальными представлениями сигналов «успех-провал» процесса независимо от существования логических значений, а также текстами диагностических сообщений
CE	Появляются понятие «предикат» и специальная функция ERROR для выбора обработчиков диагностических ситуаций и продолжения недоопределенных вычислений
CM	Появляется типизация значений и данных, а также сигнатур операций и функций, используемая для профилактики неудачных вычислений
CC	Вводятся специальные схемы управления вычислениями на основе проверки условий соответствия данных и действий для их обработки

Окончание табл. 6

Индекс позиции	Пояснение
CS	Появляются структуры с равноправным доступом к разнородным элементам, возможно, с их разметкой и указанием кратности вхождения
SV	Вводятся мультизначения и специальные значения для представления разного рода неопределенностей, раскрытие или игнорирование которых может быть полезно для продолжения вычислений
SE	Появляются операции над произвольным числом параметров, их распространение на любые структуры данных, проекции формул, отложенные вычисления, формулы доопределения и продолжения вычислений
SM	Понятие присваивания распространяется на обмен данными с переферийными устройствами. Возникает идентификация устройств, абстрактные и конкретные имена, паспорта взаимодействий, сигналы готовности действий, время ожидания отклика от устройства, копии и многое другое, отражающее специфику разного оборудования
SC	Возникают отображения, средства ввода-вывода, сетевое управление, потоки, эстафета обслуживания, итерирование, условия срабатывания, актив-пассив, администрирование, сервер, ОС
SS	Одновременность, синхронизация, взаимоисключения, сигналы и сообщения, пакеты, настройки, сервисы, конфигурации

Примечание: индекс позиции (клетки в понятийной матрице) имеет формат: XY, где X – обозначает строку понятийной матрицы, а Y – столбец, оба имеют значения из E, M, C, S.

4. Семантические системы основных парадигм (паспорт парадигмы)

Рассматривая любые семантические системы, важно отметить разницу в характере выполнения функций таких систем в различных контекстах. Так, для любого множества данных D, представляющих значения V произвольной природы, реализационно различимы схемы функций для методов вычислений E, средств доступа к памяти M через адреса N, особенностей управления вычислениями C и коммуникации или обратимой комплексации и структурирования данных S. Это приводит к представлению об основных категориях семантических систем различно реализуемых схем функций F. Исторически на уровне аппаратуры такие категории систем обладали кумулятивным эффектом в порядке «DEMCS» (см. табл. 5) – представление чисел, арифмометр, калькулятор, дифференциальный вычислитель, компьютер, причем аппаратные подсистемы могут взаимодействовать каждая с каждой. Детализация определений ЯП по семантическим и прагматическим особенностям позволяет достаточно четко различать наиболее известные парадигмы программирования, поддержанные во многих языках [5].

При подготовке программ парадигмальные различия проявляются в отношении порядка между категориями семантических систем. Переход к отладке программ влечет прагматическую разницу в упорядочении слоев, не влияющую на выбор парадигмы: или раскрутка программы от ядра через расширение и уточнение к универсальной программной системе, или сборка из готовых компонентов через настройку и адаптацию к требованиям целевого интегрированного приложения. И то, и другое может быть осуществлено на базе любого ЯиСП.

Таблица 7

Ряд категорий семантических систем аппаратного и реализационного уровня

Table 7

A number of categories of semantic systems at the hardware and implementation level

Подсистема	Примечание
Категории семантических систем	**Аппаратно ориентированная семантика вычислений**
D: данные	Данные из множества D представляют значения из V и шкалу прерываний
E: вычисления	Операции по двум-одному значению производят одно или два значения
M: память	Соответствие между адресами из множества N и хранимыми по этим адресам представлениями из множества D для значений из V допускает разные методы доступа к элементам памяти, включая замену хранимых значений, за исключением адреса 0
C: управление	Сравнение значений с нулем позволяет управлять ходом вычислений наряду с передачами по меткам и обработкой прерываний, не считая перехода по порядку
S: коммуникации	Конструирование сложных данных учитывает возможности команд адресации в памяти и использование внешней памяти
Слои реализационных решений	**Прагматика реализации уточняет методы выислений**
E: ядро	Вычисления над значениями, имеющими представление в виде данных, обрабатываемых непосредственно машинными командами, устройство которых не доступно с уровня языка программирования
M: укрупнение	Данные, результаты и команды их обратотки размещаются в памяти, что позволяет использовать их многократно и к сложным данным, включая процедуры, обращаться по адресу, подобно обращению к команде
C: граница	События, происходящие в процессе вычисслений или во внешней обстановке, могут порождать сигналы или распознаваться по специальным условиям, которые можно учитывать в программе
S: универсальность	Обстановка, в которой выполняются вычисления, может формировать любые входные данные, включая непредусмотренные ранее или неподходящие. Поэтому возникает необходимость расширение области определения программы дополнительными значениями, позволяющими различать подходящие данные и использовать разнообразные устройства

5. Пример представления результатов анализа и сравнения ЯП

Результаты анализа языка Pure Lisp представлены в табл. 8, а языка Pascal – в табл. 9. В клетках таблиц расположены фрагменты текстов программ на анализируемых языках или обозначения фрагментов или понятий языка.

Понятийная матрица позволяет средства языка Pure Lisp характеризовать в рамках семантической системы вида <V, F, R>, отображаемой в абстрактную машину SECD, обладающую независимыми регистрами [6]. Функции такой системы могут принадлежать основному множеству, и правило применения функций может входить в набор функций.

Таблица 8

Понятийная матрица определения языка Pure Lisp

Table 8

The concept matrix of the definition of the Pure Lisp language

№ строки	№ столбца	V	E	M	C	S
	Виды функций	V	F: V* → V	AL : Atom → V	(F → {NIL, T})*	A ↔ K
E	Ядро	a) NIL ; и атомы	a) eq: V V→{NIL,T}	a) AL =(… (NIL . NIL) (T . T))	a) eval:Sexpr AL→ V b) quote	a) cons: V1 V2 → (V1 . V2) b) car: (V1 . V2) →V1 c) cdr: (V1 . V2) →V2
M	Специальные функции	a) Sexpr: (V . V)	a) (lambda (x1 x2 ...) form) b) (label FN F)	a) pairlis X Y→ AL b) assoc: X AL→V	a) ((lambda (x1 x2 ...) form) v1 v2 …) b) ((label FN F) …)	a) (V . . .) ; list ; Список b) (F v1 v2 . . .) ; Форма
C	Границы	a) NIL b) «Нет переменной» c) «Нет функции»	a) atom:V→{NIL,T} b) apply: F V* AL→V c) ERROR	a) null b) Свободные переменные c) Типы значений	a) cond b) FUNCTION	a) Строки b) Clozure = Funarg
S	Общность Практичность	a) Числа b) Строки	a) evlis : (form*) → V* b) evcon : ((form form)*) → V **c) + - * /**	a) Псевдо-функции b) свойства	a) map b) reduce c) lazy	a) PRINT b) READ

Таблица 9

Понятийная матрица определения языка Pascal

Table 9

The concept matrix of the definition of the Pascal language

№ строки	№ столбца	V	E	M	C	S
	Виды функций	V	F: V* → V	M : Id→ V+F	(F → {True, False})*	V N ↔ Array [N] of V

E	Ядро	a) type = (a,b, . . .)	a) = <>
			b) pred succ
				a) Id
				b) var X : int;
				c) X := 1;
				d) label L;
				e) L: a := 1;
					a) (1*(2+X))
					b) S1 ; S2
					c) ключевые слова
					d) goto L;
						a) s: array [1 . . 9] of int;
						b) s [5] := x;
						c) x := s [5]

M	Расширение	a) false true
		b) maxint
		c) integer
		d) (1,3,6,9, . . .)
		e) const A=1;B=2;
		f) type
		g) ta = 1 .. 10;
			a) + - * / div mod
			b) F (1, x, ...)
				a) var va : ta;
				b) begin .
				c) begin . . .
				end. . end
				d) func F subr S
				e) end.
					a)if X then S1 else S2
					b) begin S1 ; S2... end
					c) S (X, Y)
					d) forward;
						a) s: array [1 . . 9]
						of array [1 . . 3] of int;
						b) s [5,2] := x;
						c) x := s [5,2]

C	Границы	a) «Нет определения» b) “не соотв. ТД :=” “не соотв. ТД операнда” “не соотв. ТД параметра” c) “выход из диапазона”	a) = < > >= <= b) F: V → {True, False} c) in d) not or and	Типы данных: a) int b) char c) array ...	a) case b) Соответствие ТД	a) record b) set c) record … case
S	Практичность	Указатель: a) Nil b) X^	a) Функция-параметр	a) Процедура-параметр, 1) C*тандартные процедуры* *2) NEW* *3) DESPOSE, ...*	a) while for until	*1) Стандартные библиотеки* *2) WRITE* *3) READ*

Понятийная матрица, характеризующая средства языка Pascal, обладает меньшей кумулятивностью. Pascal определен в рамках семантической системы вида <V; F; R>, отображаемой в пи-код [7], в котором, в отличие от SECD, регистры SEC размещены в общей памяти. Составляющие такой системы строго разделены. Множество V фиксировано по составу, неявное определение R не допускает модификаций, функции могут выполнять роль данных в терминах указателей, связь с внешним миром определяется вне языка через библиотечные модули.

Такие понятийные матрицы достаточны для представления результатов сравнительного анализа ЯП и оценки сложности их применения, например, по числу различимых конструкций в каждой клетке понятийной матрицы. Тем не менее, сравнение таких матриц требует определенных усилий. Более простое, хотя и грубое, сравнение можно выразить как в табл. 10–12.

Таблица 10

Представление результатов сравнения понятийной сложности определений языков Pure Lisp и Pascal (число пунктов в клетках табл. 7, 8)

Table 10

Presentation of the results of comparing the conceptual complexity of the definitions of the languages Perl Lisp and Pascal (the number of points in the cells of Tables 7, 8)

Lisp						Pascal					
	V	E	M	C	S		V	E	M	C	S
E	1	1	1	2	3	E	1	2	5	4	3
M	2	2	2	2	2	M	7	2	5	4	3
C	3+	3	3	2	2	C	3+	4	3	2	3
S	2	3	2	3	2	S	2	1	1/**3**	1	0/**3**

Трудоемкость сравнения таких матриц много меньше, сразу видно соотношение числа семантических сестем в разных категориях. Варианты кратких формы представления тех же самых результатов сравнения сложности ЯП (число пунктов в клетках и строках табл. 8, 9) показаны в табл. 11.

Таблица 11

Краткие формы представления результатов сравнения сложности ЯП

Table 11

Brief forms of presentation of the results of comparing the complexity of the PL

Lisp – 43+	Pascal – 56+ / 6*
(E (1 1 1 2 3) M (2 2 2 2 2) C (3+ 3 3 2 2) S (2 3 2 3 2))	(E (1 2 5 4 3) M (7 2 5 4 3) C (3+ 4 3 2 3) S (2 1 **1+3*** 1 **0+3***))
((E 8) (M 10) (C 13+) (S 12))	((E 15) (M 21) (C 15+) (S 5 /**6***))

Если то же самое представить визуальной диаграммой в стиле деловой графики, то происходит значчительное снижение трудоемкости грубого сравнения характеристик (см. табл. 12), мгновенно с одного взгляда видны различия в потенциале категорий семантических систем. На диаграмме цвет символизирует категорию семантической системы, высота столбика – число семантических систем, входящих в данную категорию. Горизонтальный прямоуголь-

ник символизирует прагматический слой реализационных решений, а размещенные сверху столбики – необходимое библиотечное сопровождение языка, обычно вызванное отсутсвием средств отладки программ. Каждой позиции визуальной диаграммы соответствует клетка понятийной матрицы.

Таблица 12

Визуальная форма представления результатов сравнения сложности ЯП

Table 12

Visual representation of the results of comparing the complexity of the PL

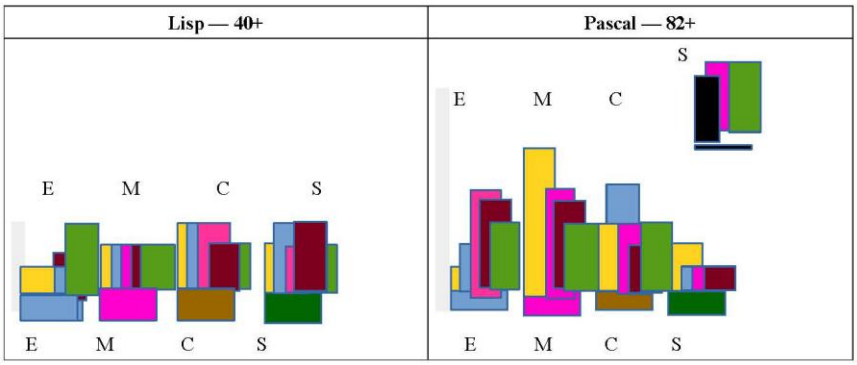
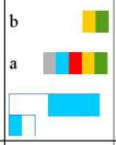
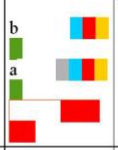


Таблица 13

Понятийная матрица определения языка Pure Lisp

Table 13

Pure Lisp Language Definition Concept Matrix

№ стр		V	E	M	C	S
		V	F: V* → V	AL : Atom → V	(F → {NIL, T})*	A ↔ K
E		a) NIL ; и атомы	a) eq: V V→{NIL,T}	a) AL =(… (NIL . NIL) (T . T))	a) eval:Sexpr AL→ V b) quote	a) cons: V1 V2 → (V1 . V2) b) car: (V1 . V2) →V1 c) cdr: (V1 . V2) →V2
M		a) Sexpr: (V . V)	a) (lambda (x1 x2 ...) form) b) (label FN F)	a) pairlis X Y→ AL b) assoc: X AL→V	a) ((lambda (x1 x2 ...) form) v1 v2 …) b) ((label FN F) …)	a) (V . . .) ; list ; Список b) (F v1 v2 . . .) ; Форма
C		a) NIL b) «Нет переменной» c) «Нет функции»	a) atom:V→{NIL,T} b) apply: F V* AL→V c) ERROR	a) null b) Свободные переменные c) Типы значений	a) cond b) FUNCTION	a) Строки b) Clozure = Funarg
S		a) Числа b) Строки	a) evlis : (form*) → V* b) evcon : ((form form)*) → V c) + - * /	a) Псевдо-функции b) свойства	a) map b) reduce c) lazy	a) PRINT b) READ

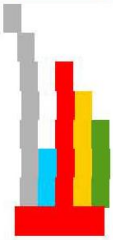
Появляется возможность с каждой понятийной матрицей сопоставить ее визуальную диаграмму, позволяющую мгновенно видеть и оценвать разнообразие семантических систем в языках и мощность отдельных категорий семантических систем по числу входящих в них элементов (табл. 13, 14).

Таблица 14

Понятийная матрица определения языка Pascal

Table 14

The concept matrix of the definition of the Pascal language

№ стр		V	E	M	C	S
		V	F: V* → V	M : Id→ V+F	(F → {True, False})*	V N ↔ Array [N] of V
E	d c b a 	a) type = (a,b, . . .)	a) = <> b) pred succ	a) Id b) var X : int; c) X := 1; d) label L; e) L: a := 1;	a) (1*(2+X)) b) S1 ; S2 c) ключевые слова d) goto L;	a) s: array [1 . . 9] of int; b) s [5] := x; c) x := s [5]
M	f e d c b a 	a) false true b) maxint c) integer d) (1,3,6,9, . . .) e) const A=1;B=2; f) type g) ta = 1 .. 10;	a) + - * / div mod b) F (1, x, ...)	a) var va : ta; b) begin . c) begin . . . end. . end d) func F subr S e) end.	a)if X then S1 else S2 b) begin S1 ; S2... end c) S (X, Y) d) forward;	a) s: array [1 . . 9] of array [1 . . 3] of int; b) s [5,2] := x; c) x := s [5,2]
C	d c b a 	a) «Нет определения» b) “не соотв. ТД :=” “не соотв. ТД операнда” “не соотв. ТД параметра” c) “выход из диапазона”	a) = < > >= <= b) F: V → {True, False} c) in d) not or and	Типы данных: a) int b) char c) array ...	a) case b) Соответствие ТД	a) record b) set c) record … case
S	3 2 1 b a 	Указатель: a) Nil b) X^	a) Функция- параметр	a) Процедура- параметр, 1) *Стандартные процедуры* 2) *NEW* 3) *DESPOSE,* ...	a) while for until	*1) Стандартные библиотеки* 2) *WRITE* 3) *READ*

Можно создать гипертекст, в котором от каждой позиции диаграммы возможен переход к соотвествующей клетке понятийной матрицы. Теперь можно понятийную матрицу линеаризовать, и каждому элементу категории семантических систем дать учебные примеры. Число необходимых примеров дает понимание сложности понятий языка (табл. 15).

Таблица 15

Начало ряда учебных примеров программ
(ФП: Pure Lisp – чистые вычисления)

Table 15

The beginning of a series of training program examples
(FP: Pure Lisp-pure computing)

Индекс семантической системы	Понятие	Примеры	Примечания
	Любую информацию можно представить символами. Информационная обработка может быть сведена к выражениям из функций с использованием небольшого числа операций над символами		Исходная гипотеза
EVa	Минимальные значения	NIL	= ( ) = пустой список и «ложь»
		T и другие атомы	Атомы без начального значения
EEa	EQ (равенство)	(EQ NIL NIL)	«истина»
		(EQ T T)	Если T имеет значение, то «истина»
		(EQ NIL T)	«ложь» или «неопределено»
ESa	CONS	(CONS NIL NIL)	Пара (NIL . NIL) = (NIL)
		(CONS x y)	Пара (x . y), **если** x и y имеют значение
ESb	CAR	(CAR (CONS x y))	x – Левый элемент пары (x . y)
ESc	CDR	(CDR (CONS x y))	y – Правый элемент пары (x . y)
EMa	Ассоциативный список	((T . T) (NIL . NIL))	Содержимое памяти, показывающее, что NIL и T означают сами себя, так как связаны сами с собой
ECa	EVAL (Вычисление)	(EVAL NIL) (EVAL T) (EVAL (CONS T NIL)) (EVAL A)	NIL T (T . NIL) = (T) **Если** A вычислимо, то его значение, **иначе** «неопределено»
ECb	QUOTE (Блокировка) ' – эквивалент	(QUOTE A) (QUOTE (EVAL A)) (EVAL (QUOTE A))	A (EVAL A) A
		'A ' (EVAL A) (EVAL ' A)	; эквивалент (QUOTE A) ; эквивалент (QUOTE (EVAL A)) ; эквивалент (EVAL (QUOTE A))

Примечание: индекс семантической системы имеет формат: XYz, где z – номер пункта в понятийной матрице, X – обозначает строку понятийной матрицы, а Y – столбец, X и Y имеют значения из E, M, C, S.

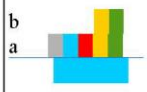
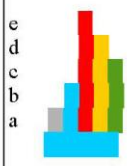
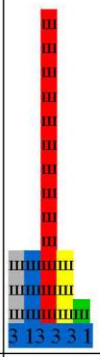
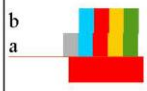
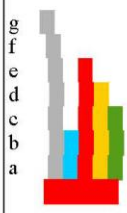
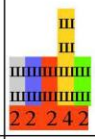
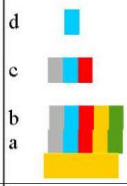
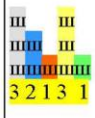
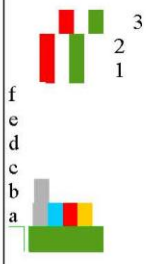
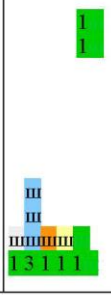
Можно создать гипертекст, в котором от каждой семантической системы из соответствующей клетки понятийной матрицы возможен переход к комплекту учебных примеров, показывающих особенности ее проявления при разработке программ. Диаграммы, характеризующие понятийные матрицы разных языков, можно использовать при их грубом сравнении, чтобы сразу видеть потенциал отдельных категорий семантических систем и быстро отбраковывать заведомо неподходящие. Известно, что в информационном поиске эффективность определяется скоростью отбраковки неподходящего (табл. 16).

Таблица 16

Сравнение ЯП

Table 16

PL comparison

	Лисп	Паскаль	Си - Имп - 52	Примечание
Ядро				Компактное ядро Лиспа нацелено на минимум используемых абстракций. Несколько более объёмное ядро языка Паскаль отражает внимание к понятности. Си опирается на большое число машинных команд.
Расширение				Паскаль поддерживает два формата расширений – отдельно для выражений и для операторов над разнообразными типами данных
Границы				Разное понимание границ вычислимости в различных категориях семантических систем
Контекст				Лисп обладает полным спекром средств для отлаки программ и связи с внешним миром. Паскаль и Си для связи с внешним миром используют библиотечные модули, без которых отладка программ проблематична.

6. Парадигмальная декомпозиция мультипарадигмальных ЯП

Для мультипарадигмальных ЯП семантическому анализу желательно предпослать парадигмальную декомпозицию, потому что поддержка средств для разных парадигм может обладать существенными прагматическими различиями, что делает их неудобными для сравнения. И наоборот, отдельные парадигмы могут быть поддержаны общими прагматическими механизмами. Пример такой декомпозиции приведен в табл. 17–20.

Таблица 17

Понятийная матрица определения языка Си (ядро – скаляр)

Table 17

The concept matrix of the definition of the C language (the kernel is a scalar)

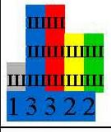
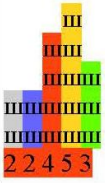
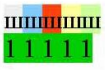
№ стр	№ столбца	V: value	E: eval	M: memory	C: control	S: structure
	Виды функций	V	F: V* → V	AL : Atom → V	(F → {NIL, T})*	A ↔ K
S		•	•	•	•	•
E		• 1 23 Const	• != Не равно == • + - • &v	• int [описатель] [имя]; • X = y • & взятие адреса	• Справа налево • i = j + 2	• [ ] • x [4]
M		• Переменные • скалярные функции	• A + 34 выражения • F (23)	• (x = y) + 1 • x [4] = 5 • i = j + 2; • int F (x) {return x = 2;}	• (y*z)/w изнутри наружу Расставлять скобки, • *Return (x);* • **F (2) + 3** • **While** • **main (argc, aigv, envp)** старт	• стек операндов • таблица переменных • таблица функций
C		• сообщения	• < > <= >= • «выход результата за пределы диапазона используемого типа данных»	• «Нет определения»	• ae ? e1 : e2 или • pe ? e1 : e2	• “библиотека недоступна”
S		• Library	• Include препроцессор ***#include <math.h>*** ***из стандартного каталога***	• Scanf (& Var)	• Printf	• <stdio>

Таблица 18

Понятийная матрица определения языка Си (расширение – процедуры)

Table 18

The conceptual matrix of the definition of the C language (extension – procedures)

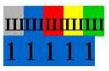
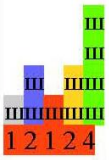
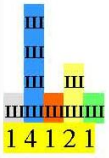
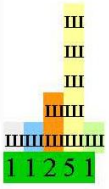
№ стр	№ столбца	V: value	E: eval	M: memory	C: control	S: structure
	Виды функций	V	F: V* → V	AL : Atom → V	(F → {NIL, T})*	A ↔ K
E		• *Список аргумен-тов*	• *type*	• *void*	• *el, e2*	• *(el, e2, . .) список параметров*
M		• Указа-тель на функцию int (*f)();	• *(F (a, b, ...) + 2) имя_функ-ции (e1, e2,.. .., eN);* • *type приведение ТД*	• *void ermesg(s) char * s; { printf("***%s\n", s); }*	• *Return 0;* • *F(выр , ...); G (3); fclose (file);*	• *Стек параметров* • *стек локалов* • *список аргументов* • *struct*
C		• *не совпада-ют ТД Типы значений*	• *Совме-стимость типов +* • *=* • *Ф (23)*	*Внешние переменные* • *extern long linfunc(); int y; y = linfunc (3.05, 4.0, 1e-3);*	• *typedef старый_тип новый_тип* • *Функция main() выполняется первой.*	• *union bigword { long bg_long; char*bg_char[4]; };*
S		• *Текст програм-мы*	• ***#include "ABC"***	• *#define идентификатор1 (идентификатор2, ...) строка* *#define abs(A) (((A) > 0) ? (A) : -(A))* • *#undef*	• if ABC + 3 • #ifdef ABC • #ifndef ABC • #else . . . • #endif	• *Описания формальных параметров main*

Таблица 19

Понятийная матрица определения языка Си (контроль – императив)

Table 19

The conceptual matrix of the definition of the C language (control – imperative)

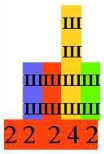
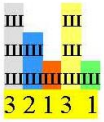
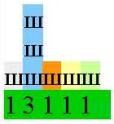
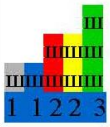
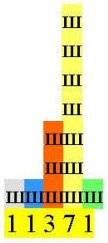
№ стр	№ столбца	V: value	E: eval	M: memory	C: control	S: structure
	Виды функций	V	F: V* → V	AL : Atom → V	(F → {NIL, T})*	A ↔ K
Линейные участки						
E		• float, 4037e-5 double • char *p; char **t; char m[7][50]; Строки "This is a character string" • указатели	• * / • *av += ae* • & • * • iv++ • iv-- • ++pv • --pv • intptr = &n; • *Использование: *pe • Адресные операции • &Использование: &v	[описатель] [имя] ([список]); Инициализация переменных *[описатель] [имя] = [инициализатор]* • *int 1 = 1;* • *int a[] = {1,4,9,16,25,36};* • {x = 1; y = 2; z = 3;}	• ({Последовательность инструкций} • (следующий оператор) • можно задать желаемый приоритет операци	• [N] 12
M		• Метки • Null	• Любое выражение, заканчивающееся точкой с запятой (;), • (указатель _на_функцию) (e1, e2,..., eN)	Глобальные переменные Автоматические переменные • register int y; • statc	• Goto ABC2; • ABC2: x = 3; • break, • continue	• Struct • Таблица меток
C		• «Нет метки» • Не существует указанного поля • «значение вместо указателя»	• n = sizeof(arname) / sizeof(int) • *Логические операции*	• переменные, помеченные ключевыми словами	if((условие)) (оператор) • else (альтернативный оператор) • if (i == 0) break; • if (x < 0) printf("negative");	• Union
S		• Адреса из других модулей	• *#define ABC 100* • *#undef ABC* • ***#line 20 "ABC"***	• *Поименованные константы*	• for	*Описания внешних переменных*

Таблица 20

Понятийная матрица определения языка Си (универс – структуры)

Table 20

The conceptual matrix of the definition of the C language (the universe – structure)

№ стр		V: value	E: eval	M: memory	C: control	S: structure
		V	F: V* → V	AL : Atom → V	(F → {NIL, T})*	A ↔ K
E		• Указатель	• *ptr = c;	• описания • {x = 1; y = 2; z = 3;}	• F (g (, , ,)) + 1 • for { for While , do while }	• Typedef • [ ] [ ] [ ] • [ ] { [ ] struct Union}
M		• Null	• *Использование: *fpe • fpe = funcname; (*fpe) (argl,arg2);	• Type F () { G () ;} • *ptr = c; • *Использование: *fpe • Значением выражения является функция, адресуемая указателем fpe. П р и м е р • fpe = funcname; (*fpe) (argl,arg2);	• F (G ( )) ; • While { for While , • do while } • do while { for While , • do while } • if((условие)) (оператор • else (альтернативный оператор) • if (i == 0) break; • if (x < 0) printf("negative") – вложенные ветвления	• Struct { [ ] struct Union} • [ ] . →
C		• «указатель – знач»	• Sizeof	• Typedef • extern • extern int Global_var; extern char *Name; extern int func();	• Switch case • case 3 + 4: Неправильно: case X + Y: • Вариант default не обязательно должен быть последним. • switch (x) { case 'A': printf("CASE A\n"); break; • case 'B': case 'C': printf("CASE B or C\n"); break; • default • вложенные переключатели	• Union { [ ] struct Union} [ ] . ->
S		• *Включаемые файлы*	• коды других модулей	• malloc(), • calloc(), • realloc()	• *extern в начале текущего файла*	• **ФАЙЛЫ**

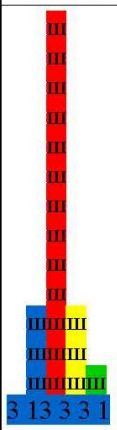
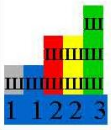
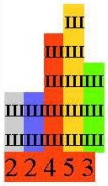
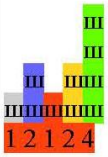
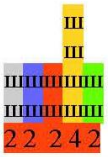
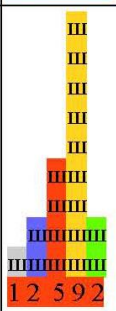
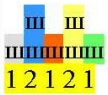
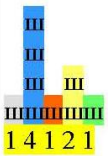
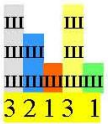
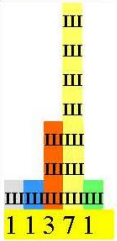
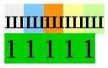
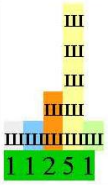
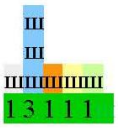
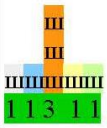
Теперь можно увидеть достаточно краткую сводку общей картины категорий семантических систем языка (табл. 21).

Таблица 21

Сравнение парадигм Си

Table 21

Comparison of C paradigms

	Скалярное – 39 систем	Процедурное – 34 системы	Императивное – 52 системы	Структурное – 48 систем
Ядро				
Расширение				
Границы				
Контекст				

Визуальные диаграммы в табл. 21 позволяют мгновенно оценить разницу в потенциале парадигм, поддержанных языком Си, подтверждающую общее мнение о его возможностях. Больше всего в языке средств для императивного программирования – это ведущая парадигма. Далее по мощности идет структурное программирование, играющее заметную роль. Скалаярное и процедурное программирование имеют поддержку, но рассматриваются как второстепенные парадигмы, меньше влияющие на практическое прешение задач, для решения

которых предназначен Си [8]. Таким образом, получается нечто вроде визитной карточки ЯиСП, позволяющей быстро выполнять их грубое сравнение.

Заключение

Изложенные представления можно рассматривать как конкретизацию понятийной сложности по Колмогорову [9]. Близкие работы начаты еще в середине 1960-х гг., когда возникла задача создания ЯП, поддерживающих процессы разработки СП при создании новых ЯиСП. В конце 1970-х гг. в рамках работ по проекту МАРС [10] был выполнен обзор динамики развития компьютерных конфигураций, что обосновывает выбор основных семантических систем уровня аппаратуры, допускающих эффективную реализацию. В те годы был выполнен ряд серьезных попыток создания языков системного программирования, позволяющих конструировать системы программирования. Наиболее продвинутые из них, такие как Венская методика, не учитывали оценок трудоемкости и понятийной сложности системного программирования, но они составили базу Си-ориентированным LEX-YACC, идеи которых унаследовали Clang-LLVM.

В настоящее время число новых языков программирования стремительно возрастает [1], что требует специальной разработки методов лаконичного представления их особенностей для практичной оценки их возможностей. Предложенную визуализацию при оценке сложности и трудоемкости программирования на базе ЯиСП можно дополнить разделением требований к постановкам задач по сферам применения на академические и производственные, а также по уровню изученности на четкие, развиваемые, усложненно трудоемкие и новые.

Список литературы

1. **Болски М. И.** Язык программирования Си. Справочник: Пер. с англ. М.: Радио и связь, 1988. 96 с.: ил. ISBN 5-256-00171-X
2. **Вирт Н.** От Модулы к Оберону // Системная информатика. Новосибирск: Наука, 1991. Вып. 1: Проблемы современого программирования. С. 63–75.
3. **Городняя Л. В.** О представлении результатов анализа языков и систем программирования // Научный сервис в сети Интернет: Тр. XX Всерос. науч. конф. (17–22 сентября 2018 г., Новороссийск). М.: ИПМ им. М. В. Келдыша, 2018.
4. **Городняя Л. В.** Систематизация парадигм программирования по приоритетам принятия решений // Электронные библиотеки. 2020. Т. 23, № 4: Тематический выпуск «Научный сервис в сети Интернет», ч. 2. С. 666–696. URL: https://elbib.ru/article/view/616/711
5. **Колмогоров А. Н**. Три подхода к определению понятия «количество информации» // Проблемы передачи информации. 1965. № 1 (1). С. 3–11.
6. **Котов В. Е.** МАРС: архитектуры и языки для реализации параллелизма // Системная информатика. Новосибирск: Наука, 1991. Вып. 1: Проблемы современого программирования. С. 174–194.
7. **Лавров С. С.** Методы задания семантики языков программирования // Программирование. 1978. № 6. С. 3–10.
8. **Мальцев А. И.** Алгоритмы и рекурсивные функции. М.: Наука, 1965. 392 с.
9. **Хендерсон П.** Функциональное программирование. М.: Мир, 1983. 349 с.
10. **Wegner P.** Concepts and paradigms of object-oriented programming. *SIGPLAN OOPS Mess*., 1990, vol. 1, no. 1, p. 7–87. DOI 10.1145

[1] См. сайт с описаниями 171 языка и 31 парадигмы: http://progopedia.ru/.

References

1. **Bolsky M. I.** The C programming language. Reference book: Trans. from English. Moscow, Radio and Communication, 1988, 96 p.: ill. (in Russ.) ISBN 5-256-00171-X
2. **Wirth N.** From Modula to Oberon. In: System Informatics. Novosibirsk, Nauka, 1991, iss. 1: Problems of modern programming, p. 63–75. (in Russ.)
3. **Gorodnya L. V.** On the presentation of the results of the analysis of programming languages and systems. In: Scientific service on the Internet: proceedings of the XX All-Russian Scientific Conference (September 17–22, 2018, Novorossiysk). Moscow, IPM named after M. V. Keldysh, 2018. (in Russ.)
4. **Gorodnyaya L. V.** Cystematizatsiz paradigms of programming by priorities of decision-making. *Electronic libraries*, 2020, vol. 23, no. 4: Thematic issue "Scientific service on the Internet", part 2, p. 666–696. (in Russ.) URL: https://elbib.ru/article/view/616/711
5. **Kolmogorov A. N.** Three approaches to the definition of the concept of "quantity of information". *Problems of information transmission*, 1965, no. 1 (1), p. 3–11. (in Russ.)
6. **Kotov V. E.** MARS: Architectures and languages for implementing concurrency. In: System Informatics. Novosibirsk, Nauka, 1991, iss. 1: Problems of modern programming, p. 174–194. (in Russ.)
7. **Lavrov S. S**. Metody zadachi semantiki yazykov programmirovaniya [Methods of setting the semantics of programming languages]. *Programmirovanie*, 1978, no. 6, p. 3–10. (in Russ.)
8. **Maltsev A. I.** Algorithms and recursive functions. Moscow, Nauka, 1965, 392 p. (in Russ.)
9. **Henderson P.** Functional programming. Moscow, Mir, 1983, 349 p. (in Russ.)
10. **Wegner P.** Concepts and paradigms of object-oriented programming. *SIGPLAN OOPS Mess*., 1990, vol. 1, no. 1, p. 7–87. DOI 10.1145

Материал поступил в редколлегию
Received
21.02.2021

Сведения об авторе

Городняя Лидия Васильевна, кандидат физико-математических наук, доцент ММФ, Новосибирский государственный университет (Новосибирск, Россия), Институт систем информатики им. А. П. Ершова СО РАН (Новосибирск, Россия)

lidvas@gmail.com

Information about the Author

Lydia V. Gorodnyaya, Candidate of Sciences (Physics and Mathematics), Associate Professor of the MMF, Novosibirsk State University (Novosibirsk, Russian Federation), A. P. Ershov Institute of Informatics Systems SB RAS (Novosibirsk, Russian Federation)

lidvas@gmail.com

УДК 004
DOI 10.25205/1818-7900-2021-19-2-53-64

Веб-платформа для фольклорных исследований на основе онтологии предметных областей

В. А. Лисин [1], **Е. А. Сидорова** [2]

[1] *Новосибирский государственный университет*
Новосибирск, Россия

[2] *Институт систем информатики им. А. П. Ершова СО РАН*
Новосибирск, Россия

Аннотация

Рассматривается веб-платформа, обеспечивающая размещение фольклорных материалов и проведение научных исследований. Фольклорные исследования связаны с изучением аудио- и видеоматериалов, фиксирующих воспроизведение элементов народного творчества на национальных языках, создание текстовых записей с переводами и комментариями на языке общего пользования (в данной работе переводы представлены на русском языке), построение картины мира на основе источников. Для структурирования и представления контента используется подход на основе онтологий, который позволяет описывать не только ресурсы, но и предметные знания в стиле Semantic Web, т. е. с помощью иерархий классов, объектов и связей между ними. Основной особенностью фольклорных исследований является необходимость синхронизации переводов (создание параллельных корпусов текстов) и разметки текстов сущностями предметной области (семантическая разметка). При этом каждый корпус сопоставляется определенной народности и имеет как свой национальный язык, так и свою уникальную систему понятий об окружающем мире. Такое представление предъявляет множество нестандартных требований к платформе, таких как работа с произвольными языками, поддержка множества онтологий, обеспечение создания и редактирования национальных предметных онтологий, семантическая разметка текстов, представление, навигация и поиск по разнородным ресурсам. Разработанная платформа предоставляет все необходимые инструменты для исследований, включая инструменты для разработки онтологий национальных предметных областей и ручного аннотирования текстов в режиме реального времени несколькими специалистами. Размещение ресурсов на платформе осуществляется на основе онтологии ресурсов, включающей такие понятия как корпус, видео- и аудиоресурсы, графическое изображение, персона, географическое место, жанр текста и т. п. Онтологии предметных областей представлены в виде иерархии, где на верхнем уровне размещается онтология универсалий, общая для всех фольклорных исследований, а наследуемые онтологии специализируются для каждого представленного национального корпуса. Веб-приложение построено на основе фреймворка Python Django и библиотеки TypeScript React, хранение данных реализовано с помощью базы данных Postgres.

Ключевые слова

онтология фольклора, редактор онтологии, инструмент разметки корпуса

Благодарности

Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта № 20-412-540001

Для цитирования

Лисин В. А., *Сидорова Е. А.* Веб-платформа для фольклорных исследований на основе онтологии предметных областей // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 53–64. DOI 10.25205/1818-7900-2021-19-2-53-64

Web Platform for Folklore Research Based on Domain Ontology

V. A. Lisin [1], **E. A. Sidorova** [2]

[1] *Novosibirsk State University*
Novosibirsk, Russian Federation
[2] *A. P. Ershov Institute of Informatics Systems SB RAS*
Novosibirsk, Russian Federation

Abstract

In this paper, we take a close look at a web platform that provides the tools necessary for working with folklore materials and conducting scientific research based on them. Folklore studies consist of working with audio and video materials, which contain the reproduction of elements of folk art in national languages, creating specific text recordings with translation and comments, written in a public language, and building a picture of the worlds based on available resources. To structure and present this content, we use an ontology-based approach, which allows linguists to describe not only the resources, but also subject knowledge in the Semantic Web style, i.e. using hierarchies of classes, objects and relationships between them. The main feature of folklore research is the need for synchronization of translations, which is achieved by creating a parallel corpora of texts, and the ability to label texts with entities of the subject area, which is called semantic markup. Moreover, each corpus is connected with a certain nationality and has both its own national language and unique system of concepts of the world around it. Such representation imposes many non-standard requirements for the platform, such as working with arbitrary languages, supporting many ontologies, ensuring the creation and editing of national subject ontologies, semantic text markup, presentation, navigation, and search across heterogeneous resources. The developed platform provides all the necessary tools for research, including tools for the development of ontologies in specific national subject areas and manual annotation of texts in real time by several specialists. Resources of the web-platform are located in the resource ontology, which includes such concepts as corpus, video resource, audio resource, graphic image, person, geographical location, genre of text, etc. Ontologies of subject areas are presented in the form of a hierarchy, where the ontology of universals, common to all folklore studies, is located at the top level. At the same time, inherited ontologies are specialized for each represented national corpus. The web application is built with Python Django framework and the TypeScript React library. Data storage is implemented using the Postgres database.

Keywords

folklor ontology, ontology editor, semantic markup tool

Acknowledgements

This work was carried out with the financial support of the Russian Foundation for Basic Research within in the framework of the scientific project no. 20-412-540001

For citation

Lisin V. A., Sidorova E. A. Web Platform for Folklore Research Based on Domain Ontology. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 53–64. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-53-64

Введение

Фольклорные исследования связаны с изучением аудио- и видеоматериалов, фиксирующих воспроизведение элементов народного творчества на национальных языках, создание текстовых записей с переводами и комментариями на языке общего пользования (в данной работе переводы представлены на русском языке), построение картины мира на основе источников. Для структурирования и представления контента в современных научных исследованиях все чаще используется подход на основе онтологий, который позволяет описывать не только ресурсы, но и предметные знания в стиле Semantic Web, т. е. с помощью иерархий классов, объектов и связей между ними. Структурированный (размеченный) контент в дальнейшем может быть использован в качестве базы для исследования того или иного аспекта языка, статистического анализа, проверки гипотез.

Для проведения подготовки и работы над материалом существует специальное программное обеспечение для создания корпусов. Функционал данных продуктов варьируется в зависимости от цели исследования и типа материала, начиная от создания коллекций коротких текстовых фрагментов и до формирования хорошо аннотированных корпусов текстов, в которых разметка включает типизацию текстовых фрагментов и выстраивание отношений ме-

жду ними на основе модели пользователя. Сама разметка может включать или не включать в себя метаданные – информацию о самом источнике (автор, дата, жанр и т. п.)

Так, система Brat [1] представляет собой веб-платформу для аннотирования текста, в частности добавления тегов к уже существующим текстовым документам. Простая типизация и классификация объектов в тексте позволяет создавать необходимые данные для распознавания именованных сущностей, бинарных отношений и простых информационных запросов. Данный сервис также поддерживает аннотации типа *n*-арных ассоциаций, которые используются для создания связей между любым числом фрагментов, участвующих в определенных отношениях. Детальное описание типов и свойств аннотаций может быть расширено использованием атрибутов, которые применимы к любому объекту корпуса. В свободном доступе также присутствуют такие приложения, как LightTag [2], TagTog [3] и Ova [4].

Несмотря на то что множество из приведенных систем поддерживает привязку фрагмента текста к нескольким меткам или возможность построения связей между метками, большинство из них не обеспечивают построение связей между фрагментами текста, поддержку нескольких разметок, комментирование или разнообразие уровней доступа к системе.

Данные веб-сервисы имеют аналогичный набор функций и один общий недостаток – их инструментария недостаточно для проведения полноценной исследовательской работы.

В данной работе рассматривается веб-платформа, обеспечивающая размещение фольклорных материалов и проведение научных исследований.

Особенности исследования фольклора

Процесс разметки, в общем случае, заключается в выделении и приписывании фрагментам текста определенных тегов. Тип разметки зависит от решаемой задачи, она может быть морфологической, синтаксической, анафорической, семантической, дискурсной и т. п. При фольклорных исследованиях помимо классических лингвистических задач, связанных с анализом языка народности, на котором представлен текст (изначально аудиозапись или видеозапись) фольклора, возникает задача выявления картины мира, основных понятий и сущностей и разметка на их основе фольклорного текста.

Одной из отличительных особенностей фольклорного ресурса является то, что источник, как правило, представлен на национальном языке. Это определяет специфику хранения и представления пользователю материалов. Каждый текстовый ресурс состоит из двух частей: текст на оригинальном языке и его перевод на язык общего пользования. При разметке или просмотре текста пользователь должен иметь возможность сразу видеть перевод для каждого значимого фрагмента, а при добавлении такого ресурса – осуществлять параллельную разметку.

Разрабатываемая система направлена на предметно-ориентированную семантическую разметку. Процесс разметки происходит на основе иерархии онтологий предметных областей. На основе размеченных данных пользователь имеет возможность проведения онтологического поиска и подтверждения результатов данного поиска фрагментами текста.

Аннотация текста включает множество «вхождений» элементов онтологии в текст или множество фрагментов размеченных элементами онтологии – экземплярами классов и отношений. Разметке подвергается только оригинал текста, переводная часть синхронизируется с оригиналом (в нашем подходе – построчно), и при необходимости ее разметка с некоторым приближением может быть получена за счет сопоставления размеченных фрагментов. Разметка отношений опирается на связующие слова и осуществляется между уже размеченными сущностями онтологии.

На рис. 1 приведен пример разметки фрагмента текста из произведения А. С. Пушкина «Евгений Онегин». В примере размечены два экземпляра сущностей «Персона» и «Река». Экземпляр «Онегин» был привязана к фрагменту текста на позиции X(1), экземпляр «Нева» – к позиции X(8). Было построено отношение между соответствующими фрагментами текста

на основе связующего слова на позиции 5. На основе этого отношения была создана связь между двумя экземплярами онтологии под названием «Родился в».

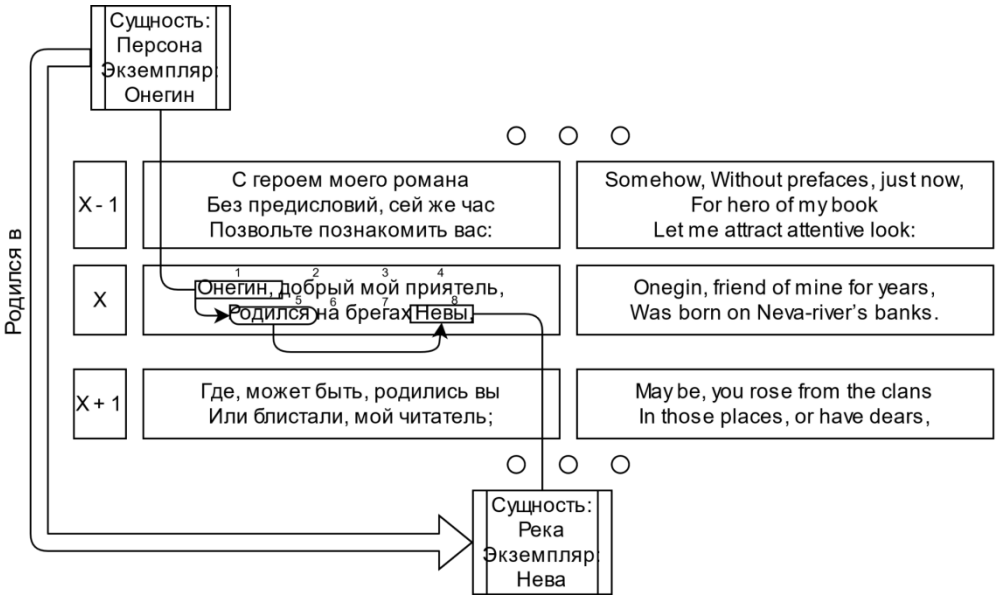


Рис. 1. Пример размеченного фрагмента текста
Fig. 1. Markup example on a text fragment

Параллельно разметке текста пользователь имеет возможность пополнить текущую онтологию ПрО сущностями, встречающимися в рассматриваемой предметной области. В роли таких сущностей могут выступать действующие лица, предметы быта, места, абстрактные понятия и т. п. При пополнении онтологии необходимо соблюдение строгого формализма, который в то же время обеспечивает гибкие средства описания предметной области, а также установления между ними семантических связей. Понятия предметной области выстраиваются в иерархию «общее – частное» с наследованием свойств по этой иерархии [5; 6]. В дальнейшем, достаточно указать требуемую единицу онтологии для ее привязки к отдельному фрагменту текста. Построение связей между сущностями ПрО подразумевает ведение списка возможных типов связей между ее объектами и классами. Такой подход позволяет специалисту формализовать множество возможных отношений, устраняя риск построения связей с похожими именами, что в дальнейшем может негативно повлиять на процедуру поиска.

Использование инструмента, основанного на web-технологиях, предоставляет специалистам возможность работы над текстом в параллельном режиме. В этом случае каждый участник процесса может создать свою версию разметки параллельного текста, не прерывая процесс работы другого участника. В ходе работы доступен просмотр прогресса сотрудника с дальнейшей возможностью объединить персональные разметки.

При разметке текста инструмент выделяет фрагменты, с которыми работал пользователь, наглядно представляя информацию об объектах и классах в тексте, связях между ними и прикрепленных к тексту ресурсах. Если разметка текста осуществляется на основе онтологии ПрО, то «внешняя» информация о ресурсе формируется на основе онтологии ресурсов. Таким образом, экземплярами онтологии ресурсов, прикрепленными к тексту, могут быть лица, проводившие предварительную обработку отображаемого материала, места записи текста, а также информация, касающаяся первоисточника материала. Онтология ресурсов была построена в соответствии со стандартом CIDOC CRM [7]. Основная роль CIDOC CRM

заключается в обеспечении обмена информацией и интеграции между разнородными источниками информации о культурном наследии. Он направлен на предоставление семантических определений и пояснений, необходимых для преобразования разрозненных или локализованных источников информации в согласованный глобальный ресурс.

Архитектура системы разметки

Для анализа возможных вариантов реализации системы были рассмотрены следующие программные средства:

Laravel представляет собой бесплатное программное средство с открытым исходным кодом для разработки веб-приложений модели MVC (табл. 1).

Таблица 1

Характеристика Lavarel

Table 1

Lavarel properties

Преимущества	Недостатки
Наличие встроенного сборщика скриптов и scss	Функционал фреймворка работает через фасады, что приводит к ошибкам IDE-систем при нахождении методов и свойств классов
Встроенный шаблонизатор	Отсутствие встроенных генераторов интерфейсов
Гибкое формирование route	Низкая производительность

Выбор Laravel обосновывается в ситуациях, когда разработчиком предъявляются особые требования к «frontend», что свидетельствует о большем вкладе средств и времени на разработку интерфейсов приложения, а также полного разделения «frontend» от «backend».

Yii является высокоэффективным компонентно-структурным PHP-фреймворком для разработки крупных приложений (табл. 2).

Таблица 2

Характеристика Yii

Table 2

Yii properties

Преимущества	Недостатки
Низкий старт разработки	Строгое форматирование route
Встроенные решения для интерфейсов	Плохое развитие
Наличие генератора моделей и контроллеров	Неразрывность «backend» и «frontend»

Выбор Yii можно осуществить в ситуациях, когда нет требований к «frontend» части и дальнейшему развитию системы и проект нужно выполнить в сжатые сроки.

Универсальный фреймворк с открытым исходным кодом Spring является лидером рейтинга популярности среди средств разработки на языке JAVA. Данное средство разработки стало широко распространенным благодаря выступлению в качестве альтернативы и замены модели Enterprise JavaBeans, предоставляя бóльшую свободу создания проектов (табл. 3).

Таблица 3

Характеристика Spring

Table 3

Spring properties

Преимущества	Недостатки
Слабо связанная система	Конфигурации в xml
Принцип инверсии управления	Сложность реализации
	Низкая производительность

Данный фреймворк специализирован на создании гибких систем, так как слабая связь его компонентов позволяет легко изменить реализацию, а инверсия управления дает возможность использования Spring в любом приложении, работающем с «spring-core». Из-за сложности проектов и конфигурации с помощью XML страдает скорость разработки и появляется необходимость использования фреймворка Spring Boot, работающего под основным фреймворком Spring.

Django появился в 2005 г. и постепенно стал одним из лучших фреймворков, который позволяет множеству разработчиков выполнять ту или иную работу в течение нескольких минут (табл. 4).

Таблица 4

Характеристика Django

Table 4

Django properties

Преимущества	Недостатки
Легкая масштабируемость	Необходимость знания всей системы для разработки
Защита от ошибок безопасности	Монолитность
Быстрота развертки проекта	Низкая производительность

Django является подходящим решением для быстрого и качественного создания проектов, но его монолитность способна ограничить рост проекта в случаях, когда инструментов недостаточно для реализации необходимого функционала.

В качестве программных средств реализации был выбран фреймворк Django.

Данное ПО находится в свободном доступе и заметно ускоряет процесс проектирования и разработки благодаря десятку дополнительных функций. В их число входят пакеты аутентификации пользователей, карты сайта, администрирования содержимого и работы с каналами. Данные модули позволяют при необходимости легко масштабировать проект.

Также, используя Django, разработчик гарантирует защиту от ошибок безопасности, ставящих под угрозу веб-приложение: кросс-сайт подлоги, инъекции, кросс-сайтовый скриптинг и т. д. Ключом данных гарантий являются отсутствие чистого SQL-кода в приложении и система пользовательской аутентификации, предустановленная в проекте.

Данный фреймворк наилучшим образом подходит для работы со средними и высокими трафиками, такими как трафики сетей средних и крупных организаций.

Набор функций, предоставляемый данным ПО, удовлетворяет требованиям, представленным в исходных данных работы, что нивелирует недостаток монолитности.

Важно отметить, что данная платформа была разработана для решения определенного набора задач в сложной и многоуровневой новостной организации. В центре набора задач стояли три важные основы:

1) предоставление возможности использования простого интерфейса для работы с базами данных, текстовым форматированием и информационными ресурсами;

2) управление UI с помощью инструментов языка разметки HTML, CSS, а также языков программирования JavaScript и его вариаций;

3) быстрая и надежная система обновления и устранения проблем работы системы в короткие сроки.

Ключом предоставления данных возможностей является разработка компонентов данных, дизайна и бизнес-логики с использованием метода «loose coupling». Данная технология обеспечивает управления блоками веб-приложения независимо друг от друга (рис. 2).

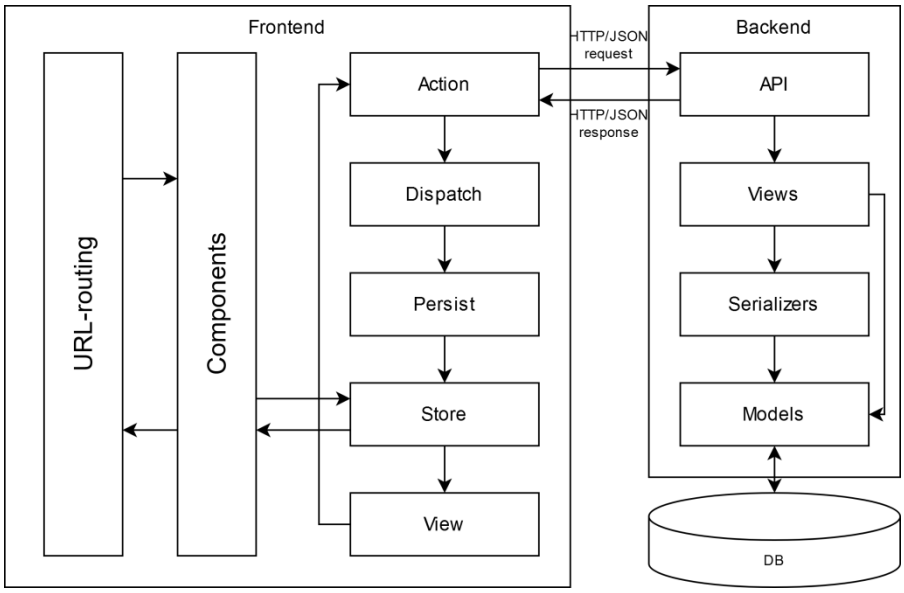


Рис. 2. Архитектура разрабатываемого приложения
Fig. 2. Architecture of the web application

Было произведено разделение логики разрабатываемого приложения на два ключевых элемента: интерфейс приложения на основе React и логику приложения на основе Django REST. Данное разделение осуществлено на основе парадигмы SoC – тип архитектуры, который позволяет разработчику проводить изменения системы в ее изолированных участках, уменьшая время простоя основного функционала сервиса.

Особенности реализации веб-платформы

Для разработки системы были установлены следующие функциональные требования:

- поддержка хранения параллельных корпусов фольклорных текстов, включающих оригинальные и переведенные тексты, а также комментарии;
- поддержка комментирования специалистами фрагментов текста в процессе его разметки;
- поддержка редактора онтологий, который позволяет вводить новые классы, их атрибуты и связи;

- поддержка хранения изображений, аудио- и видеоресурсов и их связей с разметкой текстов;
- поддержка иерархии онтологий;
- поддержка разметки корпуса текстов на основе онтологии.

Для хранения разнородной информации в базе данных Postgres был выбран пакет «db_file_storage», поддерживающий файлы, такие как текст, изображения, аудиозаписи и видеозаписи. DBFS представляет собой стороннюю библиотеку, позволяющую сохранять файлы непосредственно в базу данных.

Для просмотра текстов и отображения метаданных корпуса разработанная панель навигации обеспечивает возможность наглядного отображения дерева корпусов с их содержимым. Менеджер окон предоставляет формы для редактирования сущностей в реальном времени. На рис. 3 представлено окно с информацией об экземпляре класса «Богатырские кони». Левая часть окна содержит в себе список атрибутов данного экземпляра, наследуемый из его класса. Правая часть отображает значения данных атрибутов.

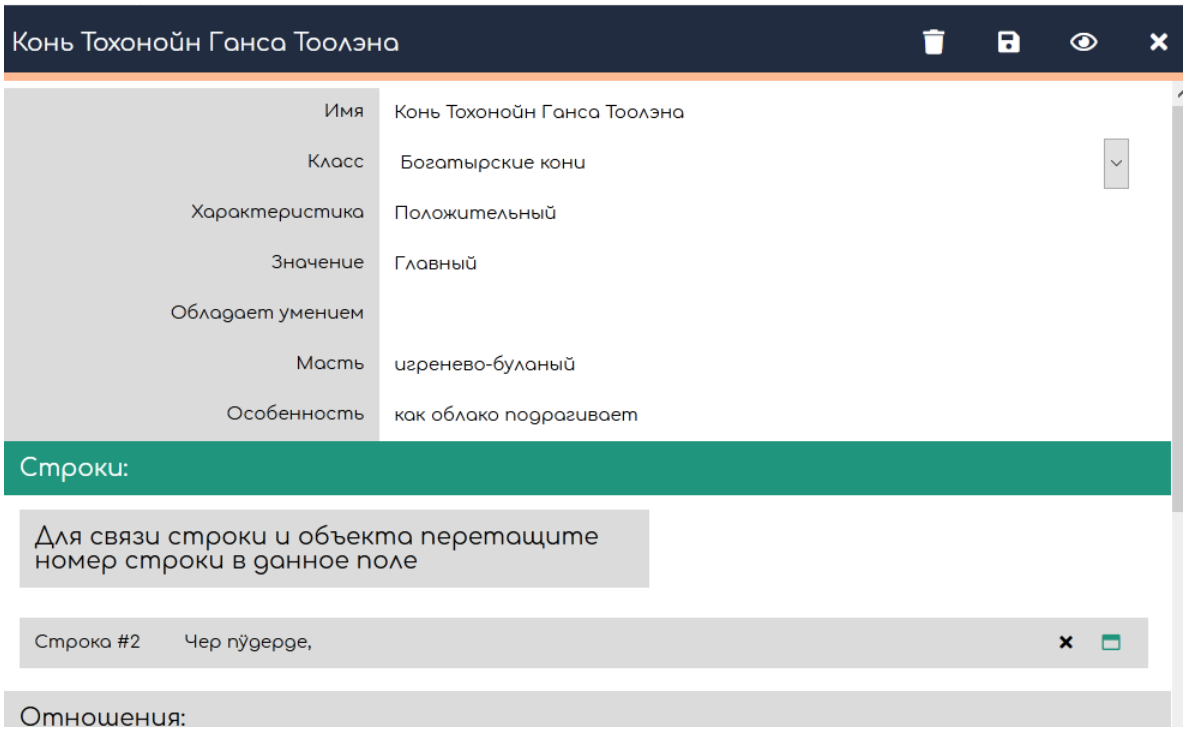


Рис. 3. Пример окна объекта онтологии предметной области
Fig. 3. Window interface example of an object in a domain ontology

Реализованная система аутентификации построена на принципе использования токенов авторизации. При регистрации нового пользователя в системе его аккаунт подтверждается администратором ресурса, так как без одобрения (активации) пользователю не будет выдан токен – зашифрованная строка, содержащая в себе имя пользователя и его пароль. Токены позволяют добавить один уровень косвенности для аутентификации. Вместо того чтобы аутентифицироваться с именем пользователя и паролем для каждого защищенного ресурса, пользователь аутентифицирует этот путь один раз (в течение ограниченного сеанса), получает ограниченный по времени токен и использует его для дальнейшей аутентификации во время сеанса. Преимущества данного метода заключаются в том, что пользователь может передать токен, как только он его получит, в другую автоматизированную систему, которой

он готов доверять в течение ограниченного времени, но не передавать данные об имени пользователя и пароль.

Функция привязки отрезков текста к объектам и классам корпуса основана на использовании инструмента «drag» и созданного менеджера окон. Инструменты по выделению созданных связей, находящихся в тексте, и отображению объектов и классов, привязанных к тексту, наглядно демонстрируют пользователю прикрепленные экземпляры и классы к текстовому ресурсу. На рис. 4 представлен фрагмент параллельного текста с выделенными в нем экземплярами классов. Зеленый маркер у номера строки сигнализирует о наличие связи данной строки с элементом онтологии. При наведении курсора мыши на данный индикатор появляется информационное окно, содержащее данные о классе и экземпляре. Также при наведении происходит выделение всех связей в тексте в соответствующих фрагментах.

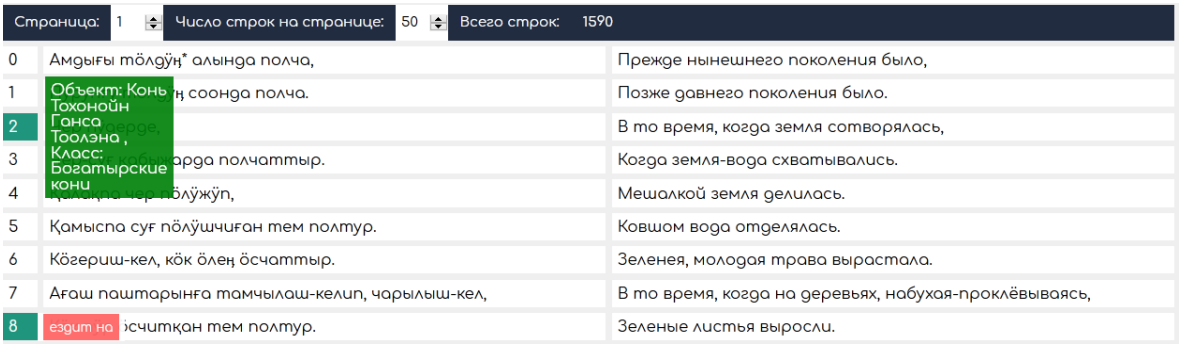


Рис. 4. Пример разметки и визуализации связи в параллельном тексте
Fig. 4. Markup and visualization example of a relation in a parallel text

В соответствии с требованиями была реализована система классов и объектов. Иерархическое дерево классов и объектов привязано к каждой корневой ветке дерева корпусов. Данный тип реализации позволяет использовать индивидуальные онтологии для каждого типа корпусов.

Для реализации поддержки связей объектов базы данных с любыми объектами была реализована архитектура на основе tag Django ORM. Данный подход заключается в использовании таблицы посредника, предоставляемой ядром фреймворка Django. Этот посредник привязан ко всем объектам в базе данных и хранит информацию об индивидуальном ключе объекта и индивидуальном ключе таблицы, к которой он привязан.

Для удобства пользователя реализована возможность добавления произвольного числа типов связей. Типы связей, типы ресурсов, авторы и места не имеют конкретной привязки к определенному корпусу для их использования во всей платформе.

Окно иерархии классов и объектов, расположенное на экране просмотра и разметки параллельного текста определенного корпуса текстов, обеспечивает быструю навигацию и выбор требуемых сущностей (классов) и экземпляров.

Инфраструктура как услуга

Учитывая малое число ожидаемых одновременных подключений, был рассмотрен вариант развертывания проекта на облачном сервисе. Окружение развертывания представляет собой среду, в которой размещается веб-приложение и обеспечивается его запуск и доступ к функционалу. Данная среда состоит из:

- аппаратных решений;
- операционной системы;

- хранилища данных;
- сервера приложений.

Несмотря на то что реализация систем возможна на собственных аппаратных решениях, общим подходом к развертыванию является применение типа удаленного доступа к вычислительным средствам под названием «Инфраструктура как Услуга» (IaaS). Большое число IaaS поставщиков позволяют выбрать предустановленные полноценные рабочие окружения, такие как Django, поддерживая его как часть своего пакета «Платформа как Услуга» (PaaS).

Были рассмотрены основные хостинг-провайдеры услуг запуска Django приложений, представленные в табл. 5. При анализе учитывались основные факторы, влияющие на выбор хостинга:

- горизонтальное и вертикальное масштабирование;
- наличие модулей анализа работы;
- географическое местоположение серверов;
- цена за соответствующий тарифный план.

Таблица 5

Хостинг-провайдеры

Table 5

Hosting providers

Название	Масштабирование	Модули	Местоположение	Цена / мес., руб.
Liquid Web	горизонтальное и вертикальное	нет	США, Россия	1300
o2switch	горизонтальное и вертикальное	да	Франция	370
MilesWeb	горизонтальное и вертикальное	нет	США, Англия	570
Heroku	горизонтальное и вертикальное	да	США, Германия, Россия	1200

В результате анализа средств для развертывания системы был выбран сервис Heroku. Функционал платформы позволяет разработчику проводить горизонтальное и вертикальное масштабирование проекта, арендуя новые виртуальные контейнеры и подключая дополнительные модули к системе, а богатая библиотека подключаемых средств наблюдения и анализа работы веб-приложений предоставляет инструменты для детального мониторинга активности. Heroku является популярным облачным инструментом «Платформа как Сервис». Благодаря развитой веб-инфраструктуре платформа устраняет необходимость в балансировки нагрузки и обслуживании серверов.

Принцип обработки Django веб-приложений заключается в работе нескольких изолированных Unix-контейнеров, организующих окружение для системы. Данные контейнеров имеют эфемерную файловую систему, проводящую самообновление и самоочищение при каждом перезапуске. Heroku, используя встроенный балансировщик нагрузок, равномерно распределяет поступающий трафик среди всех виртуальных контейнеров.

Взаимодействие с хостингом Heroku было выполнено при помощи терминала, предоставляемого платформой. Данный инструмент позволяет контролировать версии приложения, загружать и выгружать файлы конфигураций и вести историю внесенных изменений.

Для разработки приложения была использована основанная на файлах база данных SQLite. Несмотря на то что ее применение невозможно в опубликованном проекте, при развертывании приложения база данных была автоматически мигрирована в Postgres посредст-

вом инструментов, предоставляемых хостингом. В результате сервер веб-платформы по созданию и редактированию семантически размеченных корпусов был развернут на облачном сервисе с режимом доступа https://neofront.herokuapp.com/.

Заключение

В работе представлены результаты исследований, связанных с обеспечением семантической разметки фольклорных текстов на основе онтологии. Разработано веб-приложение по разметке параллельных корпусов на основе онтологий ресурсов и предметных областей. Онтологии предметных областей представлены в виде иерархии, где на верхнем уровне размещается онтология универсалий, общая для всех фольклорных исследований, а наследуемые онтологии специализируются для каждого представленного национального корпуса.

Множество нестандартных требований к платформе, такие как работа с произвольными языками, поддержка множества онтологий, обеспечение создания и редактирования национальных предметных онтологий, семантическая разметка текстов, представление, навигация и поиск по разнородным ресурсам, были обеспечены.

Разработанная платформа предоставляет все необходимые инструменты для исследований, включая инструменты для разработки онтологий национальных предметных областей и ручного аннотирования текстов в режиме реального времени несколькими специалистами. Размещение ресурсов на платформе осуществляется на основе онтологии ресурсов, включающей такие понятия, как корпус, видео- и аудиоресурсы, графическое изображение, персона, географическое место, жанр текста и т. п.

Дальнейшее развитие платформы предполагает использование графовых баз данных для размещения онтологий предметных областей и онтологии ресурсов.

Список литературы

1. **Pontus Stenetorp**, **Sampo Pyysalo**, **Goran Topić**, **Tomoko Ohta**, **Sophia Ananiadou**, **Jun'ichi Tsujii.** Brat: a Web-based Tool for NLP-Assisted Text Annotation. In: Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics, 2012, p. 102–107.
2. **Tobias Daudert.** A Web-based Collaborative Annotation and Consolidation Tool. In: Proceedings of the 12th Conference on Language Resources and Evaluation, 2020, p. 7053–7059.
3. **Juan Miguel Cejuela**, **Peter McQuilton**, **Laura Ponting**, **Steven J Marygold**, **Raymund Stefancsik**, **Gillian H Millburn.** Tagtog: interactive and text-mining-assisted annotation of gene mentions in PLOS full-text articles. *Database the Journal of Biological Databases and Curation*, 2014, no. 2014, p. bau033.
4. **Mathilde Janier**, **John Lawrence**, **Chris Reed.** OVA+: An Argument Analysis Interface. *Frontiers in Artificial Intelligence and Applications*, 2014, no. 266, p. 463–464.
5. **Кустова Г. И.**, **Ляшевская О. Н.**, **Падучева Е. В.**, **Рахилина Е. В.** Семантическая разметка лексики в Национальном корпусе русского языка: принципы, проблемы, перспективы // Национальный корпус русского языка: 2003–2005. Результаты и перспективы. М., 2005. С. 155–174.
6. **Гриневич А. А.** Онтология «Образы медвежьих песен хантов» для портала «Фольклор народов Сибири» // Вестник музыкальной науки. 2016. № 3 (13). С. 95–99.
7. **Meghini C., Doerr M.** A first-order logic expression of the CIDOC Conceptual Reference Model. *International Journal of Metadata, Semantics and Ontologies*, 2018, no. 13 (2), p. 131–149.

References

1. **Pontus Stenetorp**, **Sampo Pyysalo**, **Goran Topić**, **Tomoko Ohta**, **Sophia Ananiadou**, **Jun'ichi Tsujii.** Brat: a Web-based Tool for NLP-Assisted Text Annotation. In: Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics, 2012, p. 102–107.
2. **Tobias Daudert.** A Web-based Collaborative Annotation and Consolidation Tool. In: Proceedings of the 12th Conference on Language Resources and Evaluation, 2020, p. 7053–7059.
3. **Juan Miguel Cejuela**, **Peter McQuilton**, **Laura Ponting**, **Steven J Marygold**, **Raymund Stefancsik**, **Gillian H Millburn.** Tagtog: interactive and text-mining-assisted annotation of gene mentions in PLOS full-text articles. *Database the Journal of Biological Databases and Curation*, 2014, no. 2014, p. bau033.
4. **Mathilde Janier**, **John Lawrence**, **Chris Reed.** OVA+: An Argument Analysis Interface. *Frontiers in Artificial Intelligence and Applications*, 2014, no. 266, p. 463–464.
5. **Kustova G. I.**, **Lyashevskaya O. N.**, **Paducheva E. V.**, **Rakhilin E. V.** Semantic markup of vocabulary in the National Corpus of the Russian language: principles, problems, prospects. In: National corpus of the Russian language: 2003–2005. Results and prospects. Moscow, 2005, p. 155–174. (in Russ.)
6. **Grinevich A. A.** Ontology "Images of Khanty bear songs" for the portal "Folklore of the peoples of Siberia". *Bulletin of Musical Science*, 2016, no. 3 (13), p. 95–99.
7. **Meghini C., Doerr M.** A first-order logic expression of the CIDOC Conceptual Reference Model. *International Journal of Metadata, Semantics and Ontologies*, 2018, no. 13 (2), p. 131–149.

Материал поступил в редколлегию
Received
11.03.2021

Сведения об авторах

Лисин Владислав Александрович, студент магистратуры, Новосибирский государственный университет (Новосибирск, Россия)

vladlisin2@gmail.com

Сидорова Елена Анатольевна, кандидат физико-математических наук, старший научный сотрудник, Институт систем информатики им. А. П. Ершова Сибирского отделения Российской академии наук, Лаборатория ИИ (Новосибирск, Россия)

lsidorova@iis.nsk.su

Information about the Authors

Vladislav A. Lisin, Master's Student, Novosibirsk State University (Novosibirsk, Russian Federation)

vladlisin2@gmail.com

Elena A. Sidorova, PhD, Senior Researcher, A. P. Ershov Institute of Systems informatics of the Siberian Branch of the Russian Academy of Sciences, AI laboratory (Novosibirsk, Russian Federation)

lsidorova@iis.nsk.su

УДК 004.912 + 004.8
DOI 10.25205/1818-7900-2021-19-2-65-75

Автоматическое связывание терминов из научных текстов с сущностями базы знаний

А. А. Мезенцева [1], **Е. П. Бручес** [1, 2], **Т. В. Батура** [1, 2]

[1] *Новосибирский государственный университет*
Новосибирск, Россия

[2] *Институт систем информатики им. А. П. Ершова СО РАН*
Новосибирск, Россия

Аннотация

В настоящее время в связи с ростом научных публикаций все большую актуальность приобретают задачи, связанные с обработкой текстов научных статей. Такие тексты имеют особую структуру, лексическое и семантическое наполнение, что нужно учитывать при автоматическом анализе. Использование информации из баз знаний способно улучшить качество систем обработки текстов. Данная работа посвящена задаче связывания сущностей в текстах научных статей на русском языке, где в качестве сущностей выступают научные термины. Нами был размечен корпус научных текстов, где каждый термин связывался с сущностью из базы знаний. Также мы реализовали алгоритм связывания сущностей и протестировали его на полученном корпусе. Алгоритм состоит из двух этапов: генерация сущностей-кандидатов для входного термина и ранжирование полученного множества кандидатов. На этапе генерации список кандидатов формируется на основе построчного совпадения термина и сущности. Для ранжирования и выбора наиболее релевантной сущности для входного термина используется информация о количестве отношений сущности в базе знаний с другими сущностями, а также о количестве ссылок у сущности на другие базы знаний. Проведен анализ результатов и предложены возможные пути улучшения алгоритма, в частности использование информации о контексте термина и структуры графа знаний. Размеченный корпус выложен в открытый доступ и может быть полезен для других исследователей.

Ключевые слова

связывание сущностей, база знаний, научные термины, разметка данных

Благодарности

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 19-07-01134

Для цитирования

Мезенцева А. А., Бручес Е. П., Батура Т. В. Автоматическое связывание терминов из научных текстов с сущностями базы знаний // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 65–75. DOI 10.25205/1818-7900-2021-19-2-65-75

Automatic Linking of Terms from Scientific Texts with Knowledge Base Entities

A. A. Mezentseva [1], **E. P. Bruches** [1, 2], **T. V. Batura** [1, 2]

[1] *Novosibirsk State University*
Novosibirsk, Russian Federation

[2] *A. P. Ershov Institute of Informatics Systems SB RAS*
Novosibirsk, Russian Federation

Abstract

Due to the growth of the number of scientific publications, the tasks related to scientific article processing become more actual. Such texts have a special structure, lexical and semantic content that should be taken into account while

processing. Using information from knowledge bases can significantly improve the quality of text processing systems. This paper is dedicated to the entity linking task for scientific articles in Russian, where we consider scientific terms as entities. During our work, we annotated a corpus with scientific texts, where each term was linked with an entity from a knowledge base. Also, we implemented an algorithm for entity linking and evaluated it on the corpus. The algorithm consists of two stages: candidate generation for an input term and ranking this set of candidates to choose the best match. We used string matching of an input term and an entity in a knowledge base to generate a set of candidates. To rank the candidates and choose the most relevant entity for a term, information about the number of links to other entities within the knowledge base and to other sites is used. We analyzed the obtained results and proposed possible ways to improve the quality of the algorithm, for example, using information about the context and a knowledge base structure. The annotated corpus is publicly available and can be useful for other researchers.

Keywords

entity linking, knowledge base, scientific terms, data annotation

Acknowledgements

The study was funded by RFBR according to the research project no. 19-07-01134

For citation

Mezentseva A. A., Bruches E. P., Batura T. V. Automatic Linking of Terms from Scientific Texts with Knowledge Base Entities. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 65–75. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-65-75

Введение

Использование информации из баз знаний для решения различных прикладных задач в последнее время становится очень актуальным. Информация из базы знаний повышает качество автоматической системы, помогая разрешать лексическую неоднозначность слов и понятий, точнее определить их значение в текстах. Особую сложность представляет работа с информацией из узких предметных областей, когда подходящей терминологией владеют только специалисты. Вот почему для качественного автоматического извлечения информации важно, чтобы в системе присутствовал компонент связывания элементов текста с базой знаний.

Данная работа посвящена задаче связывания сущностей, где в качестве сущностей рассматриваются термины. Задача связывания сущностей (EL) состоит в определении упоминания сущности в неструктурированном тексте и установлении связи с сущностью в структурированной базе знаний [1].

Например, в зависимости от контекста слово «Владимир» может обозначать город в России или указывать на конкретного человека. При этом, в отличие от задачи разрешения омонимии типа сущности, в этой задаче также требуется найти конкретную сущность в базе знаний. Под базой знаний понимается база данных, содержащая структурированную информацию. Базу знаний также можно представить в виде графа знаний, в котором узлами являются сущности, а ребрами – отношения между ними. Такое представление знаний позволяет не только хранить имеющуюся информацию, но и на основе нее выводить новые факты. В качестве баз знаний традиционно выступают Википедия [1] и Викиданные [2]. Также из известных баз знаний стоит упомянуть DBPedia – база знаний, содержащая структурированную информацию, извлеченную из Википедии [2], Google Knowledge Graph [3] и Freebase (не поддерживается в настоящее время) [4]. Такие базы знаний используются в информационном поиске [5], при построении диалоговых систем [6], извлечении семантических отношений [7] и во многих других задачах.

В данной работе предложен алгоритм автоматического связывания сущностей в текстах научных статей на русском языке. В эксперименте осуществляется привязка к Викиданным, в качестве сущностей рассматриваются научные термины. Эксперимент проводился на размеченном нами корпусе, который является открытым и доступен по ссылке https://github.com/iis-research-team/ruserrc-dataset.

[1] https://wikipedia.org

[2] https://www.wikidata.org

Обзор существующих работ

Обзор методов

Традиционно задача связывания сущностей делится на 4 этапа.

Этап 1: распознавание именованных сущностей. Чаще всего этот этап выделяется в отдельную задачу, и уже выделенные сущности подаются на вход следующему этапу. Обзор методов распознавания сущностей приведен в статье [8].

Этап 2: генерация кандидатов. На этом шаге создается краткий список возможных сущностей (кандидатов) для выделенного термина. Обычно такой список создается на основании строкового совпадения (полного или частичного) упоминания в тексте с сущностями, а также применяют различные эвристики и методы для расширения этого списка (например, поиск по синонимам). Так, например, авторы статьи [9] для генерации множества кандидатов используют страницы разрешения неоднозначности и редиректов Википедии, которые в том или ином виде содержат омонимичные и синонимичные слова и фразы. Если для сущности не находится таких страниц, то используют *n*-граммы для нахождения кандидатов. Так как количество кандидатов может оказаться большим, то применяют ранжирование кандидатов: по расстоянию Джаро – Винклера [10] между сущностью и упоминанием и косинусному расстоянию между вектором контекста и вектором сущности. В финальное множество кандидатов попадают *k* первых кандидатов. В статье [11] описывается подход, который заключается в сопоставлении словоформ с заранее построенным индексом, а также применяются методы нормализации строки и меры схожести триграмм для генерации кандидатов, если ничего не было найдено по полному совпадению. Для уменьшения списка потенциальных кандидатов авторы используют априорную вероятность (на основании того, что некоторые сущности встречаются в текстах чаще, чем другие) и схожесть контекстов сущности и упоминания. Другие исследователи (например, [12]) прибегают к вычислению априорной вероятности совместной встречаемости сущности и упоминания в различных источниках: в Википедии (в заголовках страниц, в заголовках редиректов и в гиперссылках), в словаре, полученном на основе WebCorpus [3], и в словаре YAGO [4]. Максимальное значение получают те пары, которые встречаются в нескольких источниках.

Этап 3: ранжирование кандидатов. На этом шаге происходит оценка того, насколько хорошо объект-кандидат соответствует контексту. Здесь можно выделить три основных подхода. Первый подход основан на вычислении схожести контекстов, которые представляются в виде векторных представлений на основании как вручную сформированных признаков [13], так и полученных из языковых моделей [14]. При другом подходе задача ранжирования трансформируется в задачу бинарной классификации, в которой целью является определить, относится ли данное упоминание к сущности. В качестве классификатора могут использоваться наивный байесовский классификатор [15], SVM классификатор [16], глубокие нейронные сети [17]. В последнее время широкое распространение получили подходы, использующие векторные представления, полученные из графов знаний. Такая информация помогает понять, какое положение сущность занимает в графе, какими отношениями она связана с другими сущностями и др. Например, в статье [18] авторы строят векторные представления ребер графа, полученного из Dbpedia, с помощью алгоритма DeepWalk [19]. В работе [20] авторы используют алгоритм TransE [21] для векторизации сущностей в графе.

Этап 4: определение несвязанных упоминаний, для которых база знаний (KB) не содержит соответствующей сущности. Зачастую в системах этот этап отсутствует.

Больше подходов описано в работе [1], которая представляет собой обзор современных нейросетевых моделей для задачи связывания сущностей (EL). Также в данной статье отмечаются особенности данных для EL:

[3] http://www.webcorp.org.uk
[4] https://yago-knowledge.org

1) в обучающих наборах может отсутствовать даже один пример для конкретной сущности или упоминания. Для решения этой проблемы модели EL должны иметь возможности для обобщения;

2) базы знаний неполны, поэтому некоторые упоминания в тексте не могут быть правильно сопоставлены ни с одной записью в KB.

Разработано множество готовых решений, которые поддерживают английский язык и классический набор сущностей (например, OpenTapioca [22]). Библиотека DeepPavlov [5], в свою очередь, имеет предобученные модели для русского языка. Алгоритм состоит из следующих компонентов:

1) выделенная NER-моделью подстрока подается на вход в TfidfVectorizer, и получившийся разреженный вектор преобразуется в плотный;

2) Faiss-библиотека используется для нахождения *k* ближайших соседей для tf-idf векторов в матрице, где строки соответствуют tf-idf векторам слов в заголовках сущностей;

3) сущности ранжируются по числу отношений в Викиданных (количество исходящих ребер узлов в графе знаний);

4) BERT (English) или BERT (Russian) используется для ранжирования сущностей по описанию и по контексту, в котором упоминается сущность.

Обзор корпусов

Существуют корпуса для оценивания систем, решающих задачу связывания сущностей. Между собой они отличаются по нескольким важным аспектам:

1) база знаний, на которой они основаны (Википедия, WikiNews, Yago, DBpedia);

2) поддерживаемые языки (в большинстве случаев – английский, DBpedia Abstracts [23] – 7 языков);

3) источник текстов: твиты [24], новостные статьи [25]);

4) типы размеченных сущностей (большинство – классические PER, LOG, ORG; WikiMed [26] – сущности, связанные с медициной).

Для английского языка удалось найти два корпуса, в которых размечены научные термины: STEM-ECR [27] и SemEval 2015 Task 13 [28]. Насколько нам известно, в настоящее время для русского языка не существует подобного корпуса научных текстов с разметкой связанных сущностей.

Разметка корпуса

Описание разметки

Для русского языка существует корпус научных статей RuSERRC, в котором размечены термины и отношения между ними [29]. Мы дополнили этот корпус разметкой – связали выделенные термины с сущностями в Викиданных. Это свободная, совместно наполняемая, многоязычная, вторичная база данных, в которой собрана структурированная информация для обеспечения поддержки Википедии, Викисклада, а также других вики-проектов. Данная база знаний состоит из:

1) элементов, каждый из которых имеет уникальный идентификатор с префиксом Q и числовой частью, как, например, Дуглас Адамс (Q42).

2) утверждений, которые идентифицируются кодом, имеющим префикс P и числовую часть, например, учебное заведение (P69).

3) ссылки на сайты (Sitelinks) связывают каждый элемент с соответствующими ему статьями во всех клиентских вики, таких как Википедия, Викиучебник и Викицитатник.

Данный корпус содержит разметку не только терминов, но и вложенных в них сущностей, например: [*самосогласованное* [*электрическое поле*]]. При разметке связывания сущностей

[5] https://deeppavlov.ai

мы двигались от самой «большой» сущности к более «мелким» вложенным, т. е. если для самого первого уровня сущность была найдена в базе знаний, то вложенные сущности не размечаются.

Для поиска терминов в графе знаний мы допускали следующие видоизменения сущностей.

1. Все извлеченные сущности ищутся в базе знаний в нормализованной форме с учетом согласования и без учета регистра, например: *Линейных уравнений → линейное уравнение*.

2. Если из текста была извлечена сущность, подходящая по шаблону «общее понятие + название» (например, *язык программирования Python*, *операционная система Windows*), при этом в базе знаний находится только сущность с названием (например, *Python* (Q28865)), то такие две сущности связываются.

3. Если в тексте сущность написана с опечаткой, то в графе знаний мы ищем сущность без опечатки, например: *3Dреконструкцию → 3d реконструкция*.

4. Допускается поиск синонима сущности в базе знаний (проверяется запросом в поисковую систему или Википедию), например: *статистическая зависимость → корреляция*, *генетическая последовательность → нуклеотидная последовательность*, также допускается поиск перевода сущности, например, на английском языке.

5. Допускаются трансформации вида *архитектура системы → системная архитектура*.

6. Расшифрованные аббревиатуры, например: *wps → Wi-Fi Protected Setup*.

7. Если две и более сущности представлены как набор однородных членов с одним общим элементом, то каждый однородный член с общим элементом рассматривается как сущность, например: *спутниковая и мобильная связь → спутниковая связь*, *мобильная связь*.

8. Разного рода кореференции также связываются с одной сущностью, например: если в начале текста упоминается *метод k-means*, а затем в тексте *предложенный* [*метод*], то эти две сущности следует связать одним идентификатором.

9. Также мы считаем синонимами термины *подход* и *метод*.

Мы связывали термины только с сущностями в Википедии, эти сущности имеют идентификатор с префиксом «Q», в отличие от отношений, которые имеют идентификатор с префиксом «P». Также мы не связывали термины с сущностями, которые имеют тип «Научная статья».

Каждая сущность была размечена двумя асессорами. Мера согласованности была рассчитана как отношение количества сущностей без конфликта в разметке к общему количеству сущностей в корпусе и составила 82,33 %.

Наблюдения и выводы

Всего в корпусе выделено 3 386 терминов, 1 337 из которых удалось связать с сущностями в Викиданных. Средняя длина связанной сущности – 1,55 токена, минимальная длина – 1 токен, максимальная – 8 токенов.

Так как Викиданные является свободной и открытой базой знаний, то в ней встречаются ошибки, повторения сущностей и другие случаи, усложняющие работу с ней. Так, например, сущности *столбчатая диаграмма* и *гистограмма* являются в базе знаний двумя разными сущностями. Более того, есть дублирующиеся случаи, например *математический анализ* представлен сущностями Q7754 и Q149972, по описанию которых можно сделать вывод, что подразумевается одна и та же сущность. В таких случаях в разметке мы отдавали предпочтение той сущности, которая содержит больше информации, Этот показатель можно рассчитать по количеству отношений с другими сущностями, а также по наличию ссылок на эту же сущность в других базах знаний.

В Викиданных может встретиться искомая словоформа или фраза, но которая имеет смысл, отличный от того, что подразумевается в конкретном контексте. Такие сущности мы не связывали.

Вложенная сущность может иметь совершенно другое значение вне контекста: [*комплексный анализ*] [*поэтических текстов*] – в данном контексте *комплексный анализ* – это анализ текста на всех уровнях языка (об этом далее и идет речь в статье), но вне контекста термин *комплексный анализ* означает *раздел математического анализа, в котором рассматриваются и изучаются функции комплексного аргумента*.

Описание алгоритма

Нами был реализован алгоритм связывания сущностей и проверена его работа на корпусе с размеченными научными терминами. Поскольку нам не удалось найти подобных экспериментов для научных текстов на русском языке, то описанный алгоритм, скорее, является базовым и может служить отправной точкой для дальнейших исследований в данной области.

В качестве входных данных алгоритму подаются последовательность или единичный токен, соответствующий термину. Далее выполняются два основных шага: создание массива кандидатов для связывания, нахождение наиболее подходящей сущности в полученном множестве кандидатов.

Перед этапом генерации кандидатов входная строка проходит предварительную обработку – лемматизацию и приведение в нижний регистр. Здесь важно лемматизировать не отдельные слова, а сохранить согласование, например, из *обработке текстов* нужно получить *обработка текстов*. Для этого мы использовали библиотеку для анализа текстов на русском языке Natasha [6], которая позволяет приводить к нормальной форме не только отдельные словоформы, но и словосочетания, она также неплохо работает с русским языком и его лингвистическими особенностями. Стоит отметить, что данная библиотека приводит словоформу или фразу к начальной форме, сохраняя число, т. е. если грамматическая форма термина была во множественном числе, то оно сохранится, например: *мобильных приложений → мобильные приложения*. Также ошибка может возникнуть в результате омонимии, например: *у* [*приложения*] будет приведено к начальной форме *приложения*, так как на вход подается только сущность, без контекста.

На этапе генерации кандидатов входная строка сравнивается с названием сущности и ее синонимами. Если есть совпадение, то сущность добавляется в список кандидатов.

На этапе ранжирования кандидатов мы используем информацию о количестве ссылок у сущности на другие базы знаний и количестве отношений данной сущности с другими сущностями. Гипотеза состоит в том, что чем больше сущность наполнена информацией, тем более релевантной она является. Таким образом, выбор сущности для входного термина определяется по следующей формуле:

$$linked\_entity = max\ (f\ (ent_1),\ \ldots,\ f\ (ent_n)),$$

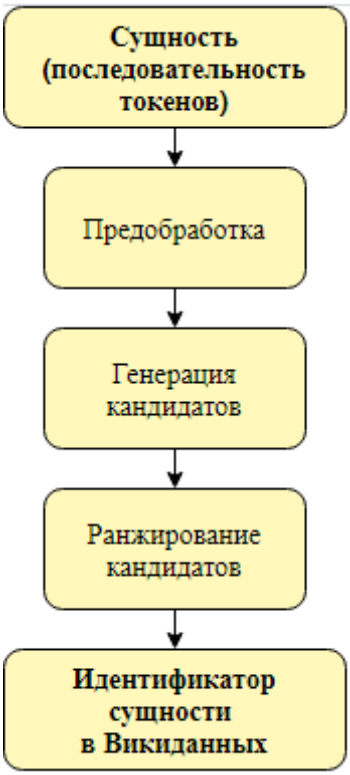
где

n – количество сущностей в полученном множестве кандидатов,

$f\ (ent_i) = numL_{ent} + numR_{ent}$, где $numL_{ent}$ – количество ссылок на другие базы знаний для данной сущности; $numR_{ent}$ – количество отношений данной сущности с другими сущностями в базе знаний.

Результатом работы алгоритма является идентификатор элемента Викиданных для входного упоминания. Общая схема работы алгоритма выглядит следующим образом:

[6] https://github.com/natasha/natasha



Следует отметить, что этот алгоритм не подразумевает использования информации из контекста, а также положения сущности в графе знаний (например, какие отношения она имеет). Добавление такой информации в алгоритм может существенно повысить качество. Кроме этого, качество алгоритма можно повысить за счет генерации синонимов и альтернативных написаний сущности для поиска кандидатов, что также пока не реализовано.

Тестирование

Алгоритм тестировался на размеченном нами корпусе (см. раздел Разметка корпуса). Для оценки качества нашего алгоритма мы использовали следующие метрики.

Точность – определяется как отношение количества верно связанных терминов ко всем терминам. Так как нам удалось связать не все термины в корпусе, информативнее будет разделить эту метрику на две: ***accuracy*** – принимает во внимание все сущности, и ***linked_accuracy*** – считается только на том наборе терминов, для которых нашлась сущность в графе знаний в корпусе. Таким образом, ***accuracy*** вычисляется по формуле

$$accuracy = n\_correct\_entities / all\_entities,$$

где

n_correct_entities – количество верно связанных терминов;

all_entities – количество всех терминов в корпусе.

Обозначим *n_all_linked_entities* количество всех терминов в корпусе, которые имеют связь с сущностью в Викиданных. Тогда ***linked_accuracy*** вычисляется по формуле

$$linked\_accuracy = n\_correct\_linked\_entities / n\_all\_linked\_entities,$$

где *n_correct_linked_entities* – количество верно связанных терминов среди всех связанных терминов.

Среднее количество кандидатов. Эта метрика показывает, насколько хорошо работает этап генерации кандидатов: если значение относительно мало, то можно улучшить алгоритм – например, также рассматривать синонимы, переводы, альтернативные написания сущностей и др. Если значение, наоборот, велико, то это может вызвать сложности при ранжировании кандидатов. Эту метрику мы также разбили на две: ***averaged_candidates*** – среднее количество кандидатов для всех сущностей, ***linked_averaged_candidates*** – среднее количество кандидатов для набора терминов, которые удалось связать.

$$averaged\_candidates = \frac{\sum_1^n |Candidates_i|}{n\_all\_entities},$$

где

$Candidates_i$ – множество полученных кандидатов для сущности;

n_all_entities – количество всех терминов в корпусе.

Обозначим *n_all_linked_entities* количество всех терминов в корпусе, которые имеют связь с сущностью в Викиданных, и $Linked\_candidates_i$ – множество сгенерированных кандидатов для всех терминов, связанных с Викиданными. Тогда формула для метрики ***linked_averaged_candidates*** имеет вид

$$linked\_averaged\_candidates = \frac{\sum_1^n |Linked\_candidates_i|}{n\_all\_linked\_entities}.$$

Наличие подходящего кандидата в списке, найденном алгоритмом (*top_candidates*). Высокое значение данной метрики показывает, что алгоритм генерации кандидатов работает с хорошей полнотой. При этом если данная метрика существенно выше точности, то это означает, что алгоритм ранжирования нуждается в доработках, так как нужная сущность имеется в списке кандидатов, но не получает наивысшего приоритета при ранжировании. Данная метрика считалась только для множества терминов в корпусе, которые имеют связь с сущностью из графа знаний, и вычислялась по формуле

$$top\_candidates = num\_correct\_sets \,/\, n\_all\_linked\_entities,$$

где *num_correct_sets* – количество множеств кандидатов для терминов, которые входят во множество *n_all_linked_entities*, содержащих верную сущность.

Полученные метрики для baseline-алгоритма:

Название метрики	Значение метрики
accuracy	0.66
linked_accuracy	0.38
averaged_candidates	1.06
linked_averaged_candidates	1.79
top_candidates	0.48

Довольно низкое значение метрики *linked_accuracy* показывает, что большая доля связанных терминов имеет форму, отличную от сущностей в базе знаний, Это означает, что этап генерации кандидатов требует доработок – нужно генерировать синонимы и другие возможные виды написания терминов. Значение метрики *top_candidates* выше значения метрики *linked_accuracy*. Это говорит о том, что алгоритм ранжирования не всегда работает корректно – здесь нужно учитывать не только наполненность информацией сущности, но и принимать во внимание контекст, в котором находится термин, чтобы сделать наиболее точный выбор. Все эти задачи мы планируем реализовать в ходе нашей дальнейшей работы.

Заключение

В ходе работы мы разметили корпус, состоящий из научных статей на русском языке, указав в нем для каждого выделенного термина его идентификатор из Викиданных. Также нам удалось реализовать алгоритм и получить метрики для данного корпуса. В дальнейшем мы планируем улучшать имеющийся алгоритм за счет использования иных подходов на каждом из этапов. При генерации важно учитывать контекст, чтобы корректно соотносить многозначные термины, а ранжирование осуществлять, используя векторные представления не только из моделей машинного обучения, но и с помощью графов.

Кроме того, мы столкнулись с тем, что базы знаний требуют обновления и дополнения, так как периодически появляются новые термины, а старые могут быть недостаточно или ошибочно заполнены. Наша система могла бы помочь в выделении из текстов необходимых сущностей и дополнении ими базы знаний в полуавтоматическом режиме.

Список литературы / References

1. **Sevgili O.**, **Shelmanov A.**, **Arkhipov M.**, **Panchenko A.**, **Biemann C.** Neural Entity Linking: A Survey of Models Based on Deep Learning. 2020. arXiv:2006.00575.
2. **Lehmann J.**, **Isele R.**, **Jakob M.**, **Jentzsch A.**, **Kontokostas D.**, **Pablo N. Mendesf**, **Hellmann S.**, **Morsey M.**, **Patrick van Kleef**, **Auer S.**, **Bizer C.** DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 2015, vol. 6, no. 2, p. 167–195. DOI 10.3233/SW-140134
3. **Dong X.**, **Gabrilovich E.**, **Heitz G.**, **Horn W.**, **Lao N.**, **Murphy K.**, **Strohmann T.**, **Sun S.**, **Zhang W.** Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In: Proceedings of SIGKDD, 2014, p. 601–610.
4. **Bollacker K.**, **Evans C.**, **Paritosh P.**, **Sturge T.**, **Taylor J.** Freebase: A collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. Vancouver, British Columbia, Canada, 2008, p. 1247–1249. DOI 10.1145/1376616.1376746
5. **Otegi A.**, **Arregi X.**, **Ansa O.**, **Agirre E.** Using knowledge-based relatedness for information retrieval. *Knowledge and Information Systems*, 2015, vol. 44, p. 689–718. DOI 10.1007/s10115-014-0785-4
6. **Shalaby W.**, **Arantes A.**, **GonzalezDiaz T.**, **Gupta C.** Building chatbots from large scale domain-specific knowledge bases: challenges and opportunities. In: Proceedings of the International Conference on Prognostics and Health Management, 2019, p. 1–8.
7. **Tripodi I.**, **Boguslav M.**, **Hailu N.**, **Hunter L.** Knowledge-base-enriched relation extraction. In: Proceedings of the Sixth BioCreative Challenge Evaluation Workshop. Bethesda, MD USA, 2018, vol. 1, p. 163–166.
8. **Li J.**, **Sun A.**, **Han R.**, **Li C.** A Survey on Deep Learning for Named Entity Recognition. In: IEEE Transactions on Knowledge and Data Engineering, 2020, p. 1–20. DOI 10.1109/TKDE.2020.2981314.
9. **Fang Z.**, **Cao Y.**, **Li Q.**, **Zhang D.**, **Zhang Z.**, **Liu Y.** Joint entity linking with deep reinforcement learning. In: The World Wide Web Conference, WWW'19. New York, NY, USA, ACM, 2019, p. 438–447.
10. **Winkler W. E.** String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In: Proceedings of the Section on Survey Research Methods. American Statistical Association, 2020, p. 354–359.
11. **Zwicklbauer S.**, **Seifert Ch.**, **Granitzer M.** Robust and collective entity disambiguation through semantic embeddings. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2016, p. 425–434. DOI 10.1145/2911451.2911535

12. **Cao Y.**, **Hou L.**, **Li J.**, **Liu Z.** Neural collective entity linking. In: Proceedings of the 27th International Conference on Computational Linguistics. Santa Fe, New Mexico, USA, 2018, p. 675–686.
13. **Bunescu R. C.**, **Pasca M.** Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, 2006, p. 9–16.
14. **Yin X.**, **Huang Y.**, **Zhou B.**, **Li A.**, **Lan L.**, **Jia Y.** Deep Entity Linking via Eliminating Semantic Ambiguity With BERT. *IEEE Access*, 2019, vol. 7, p. 169434–169445. DOI 10.1109/ACCESS.2019.2955498
15. **Varma V.**, **Pingali P.**, **Katragadda R.**, **Krishna S.**, **Ganesh S.**, **Sarvabhotla K.**, **Garapati H.**, **Gopisetty H.**, **Reddy V.B.**, **Reddy K.**, **Bysani P.** IIIT Hyderabad at TAC 2009. In: Proceedings of Text Analysis Conference, 2009, p. 102–114.
16. **Zhang W.**, **Su J.**, **Tan C. L.**, **Wang W. T.** Entity linking leveraging: Automatically generated annotation. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), 2010, p. 1290–1298.
17. **Huang H.**, **Heck L.**, **Ji H.** Leveraging deep neural networks and knowledge graphs for entity disambiguation. 2015. arXiv:1504.07678.
18. **Parravicini A.**, **Patra R.**, **Bartolini D.**, **Santambrogio M.** Fast and Accurate Entity Linking via Graph Embedding. In: Proceedings of the 2nd Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), 2019, p. 1–9. DOI 10.1145/3327964.3328499
19. **Perozzi B.**, **Al-Rfou R.**, **Skiena S.** DeepWalk: Online Learning of Social Representations. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, p. 701–710. DOI 10.1145/2623330.2623732
20. **Nedelchev R.**, **Chaudhuri D.**, **Lehmann J.**, **Fischer A.** End-to-End Entity Linking and Disambiguation leveraging Word and Knowledge Graph Embeddings. 2020. arXiv:2002.11143.
21. **Bordes A.**, **Usunier N.**, **Garcia-Duran A.**, **Weston J.**, **Yakhnenko O.** Translating Embeddings for Modeling Multi-relational Data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, 2013, vol. 2, p. 2787–2795.
22. **Delpeuch A.** OpenTapioca: Lightweight Entity Linking for Wikidata. 2019. arXiv:1904.09131.
23. **Brümmer M.**, **Dojchinovski M.**, **Hellmann S.** DBpedia Abstracts: A Large-Scale, Open, Multilingual NLP Training Corpus. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), 2016, p. 3339–3343.
24. **Noullet K.**, **Mix R.**, **Farber M.** KORE 50DYWC: An Evaluation Data Set for Entity Linking Based on DBpedia, YAGO, Wikidata, and Crunchbase. In: Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), 2020, p. 2389–2395.
25. **Minard A.**, **Speranza M.**, **Urizar R.**, **Altuna B.**, **Marieke van Erp**, **Schoen A.**, **Chantal van Son.** MEANTIME, the NewsReader Multilingual Event and Time Corpus. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), 2016, p. 4417–4422.
26. **Vashishth S.**, **Joshi R.**, **Dutt R.**, **Newman-Griffis D.**, **Rose C.** MedType: Improving Medical Entity Linking with Semantic Type Prediction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2020, p. 229–240.
27. **D'Souza J.**, **Hoppe A.**, **Brack A.**, **Yaser Jaradeh M.**, **Auer S.**, **Ewerth R.** The STEM-ECR Dataset: Grounding Scientific Entity References in STEM Scholarly Content to Authoritative Encyclopedic and Lexicographic Sources. In: Proceedings of the 12th Language Resources and Evaluation Conference, 2020, p. 2192–2203.
28. **Moro A.**, **Navigli R.** SemEval-2015 Task 13: Multilingual All-Words Sense Disambiguation and Entity Linking. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), 2015, p. 288–297. DOI 10.18653/v1/S15-2049

29. **Bruches E.**, **Pauls A.**, **Batura T.**, **Isachenko V.** Entity Recognition and Relation Extraction from Scientific and Technical Texts in Russian. In: Proceedings of the Science and Artificial Intelligence Conference, 2020, p. 41–45. DOI 10.1109/S.A.I.ence50533.2020.9303196

Материал поступил в редколлегию
Received
01.03.2021

Сведения об авторах

Мезенцева Анастасия Алексеевна, студентка, Новосибирский государственный университет (Новосибирск, Россия)

anastasiamez@mail.ru

Бручес Елена Павловна, аспирантка, Институт систем информатики им. А. П. Ершова СО РАН (Новосибирск, Россия); ассистент, Новосибирский государственный университет (Новосибирск, Россия)

bruches@bk.ru

Батура Татьяна Викторовна, кандидат физико-математических наук, старший научный сотрудник, Институт систем информатики им. А. П. Ершова СО РАН (Новосибирск, Россия); доцент, Новосибирский государственный университет (Новосибирск, Россия)

tatiana.v.batura@gmail.com
ORCID 0000-0003-4333-7888

Information about the Authors

Anastasia A. Mezentseva, Student, Novosibirsk State University (Novosibirsk, Russian Federation)

anastasiamez@mail.ru

Elena P. Bruches, PhD Student, A. P. Ershov Institute of Informatics Systems SB RAS (Novosibirsk, Russian Federation); Assistant, Novosibirsk State University (Novosibirsk, Russian Federation)

bruches@bk.ru

Tatiana V. Batura, PhD in Physics and Mathematics, Senior Researcher, A. P. Ershov Institute of Informatics Systems SB RAS (Novosibirsk, Russian Federation); Associate Professor, Novosibirsk State University (Novosibirsk, Russian Federation)

tatiana.v.batura@gmail.com
ORCID 0000-0003-4333-7888

УДК 004.9:519.178:316.35
DOI 10.25205/1818-7900-2021-19-2-76-91

Модели импорта данных из Твиттера

В. А. Попов, А. А. Чеповский

Национальный исследовательский университет «Высшая школа экономики»
Москва, Россия

Аннотация

Описываются алгоритм импорта данных из социальной сети Twitter и построение взвешенных социальных графов. Для импорта данных за основу берутся заданные посты, скачиваются пользователи, имевшие с ними какое-либо из зафиксированных взаимодействий. Далее алгоритм ориентируется на заданную конфигурацию и по ней вычисляет веса на ребрах полученного графа. Конфигурация учитывает тип взаимодействия пользователей между собой. Авторы вводят понятие (F, L, C, R)-модели информационного взаимодействия.

Авторы описывают разработанный алгоритм и реализованное программное обеспечение для построения взвешенных графов. В статье показано применение алгоритма и трех моделей на примере как отдельного поста, так и серии постов.

Ключевые слова

анализ социальных сетей, импорт данных из социальных сетей, модели информационного взаимодействия, выделение сообществ

Благодарности

Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта № 19-07-00806

Для цитирования

Попов В. А., *Чеповский А. А.* Модели импорта данных из Твиттера // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 76–91. DOI 10.25205/1818-7900-2021-19-2-76-91

Twitter Data Import Models

V. A. Popov, A. A. Chepovskiy

National Research University Higher School of Economics
Moscow, Russian Federation

Abstract

In this paper, the authors describe an algorithm for importing data from the social network Twitter and building weighted social graphs. To import data, the given posts are taken as a basis, users who have had any of the recorded interactions with them are downloaded. Further, the algorithm focuses on the given configuration and uses it to calculate the weights on the edges of the resulting graph. The configuration takes into account the type of user interaction with each other. The authors introduce the concept of (F, L, C, R)-model of information interaction.

The authors describe the developed algorithm and implemented software for constructing weighted graphs. The paper shows the application of the algorithm and three models on the example of both a single post and a series of posts.

Keywords

social network analysis, import of data from social networks, information interaction models, community detection

Acknowledgements

The study was funded by RFBR according to the research project no. 19-07-00806

For citation

Popov V. A., Chepovskiy A. A. Twitter Data Import Models. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 76–91. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-76-91

Введение

В последнее время социальные сети вошли в повседневную жизнь многих людей. Ежесекундно в сети генерируется огромный объем информации, передаются десятки тысяч сообщений и создаются сотни тысяч постов и комментариев. Все пользователи в социальных сетях имеют различные взаимодействия между собой, создавая единый социальный граф и множество потоков распространения информации. Изучение данных структур интересно для понимания действий конкретных людей и общества в целом, многие рекомендательные и рекламные системы работают на основе обработки данных о пользователях и их действиях в социальных сетях. Государственные структуры также занимаются изучением информационного потока в сетях для обеспечения безопасности. Поэтому тема изучения графов взаимодействующих объектов, в том числе социальных графов, формируемых на основе данных из социальных сетей, является актуальной в наши дни [1–4].

Одним из важных методов анализа социальных графов является подход к выявлению групп общения пользователей путем выделения на основе алгоритмов обработки графов неявных сообществ [5; 6]. Для большей эффективности данной операции требуется использовать взвешенный социальный граф, основанный на совершенных действиях пользователей. В этой связи возникает задача разработки алгоритмов для построения таких графов на основе особенностей каждой отдельной социальной сети.

Взвешенные графы

Твиттер (Twitter) – социальная сеть, в которой у каждого пользователя есть своя страница с краткой информацией. На ней пользователи могут размещать тексты до 280 символов – постить твиты (Tweets). Также есть возможность прикрепить фото или видео в постах. Каждый пользователь имеет свое уникальное имя (Username). При создании нового аккаунта каждому присваивается имя из случайных латинских букв и цифр длиной в 15 символов, которое впоследствии можно изменить на любое не занятое. Каждый пользователь может подписаться на других, т. е. читать их посты в своей ленте. Таким образом, у каждого пользователя формируется два списка: список Читателей (Followers) и список Читаемых (Following).

Все посты-твиты имеют уникальный числовой идентификатор (id), и у каждого пользователя есть возможность совершить три действия над ними: поставить отметку «Нравится» (Like), поделиться постом у себя на странице (Retweet) и написать комментарий (Reply). Каждый комментарий имеет такую же структуру, как и основной пост, т. е. считается твитом.

Действия пользователей фиксируются на их личных страницах. В главном разделе находятся все твиты и ретвиты этого пользователя, в разделе «Твиты и ответы» (Tweets & replies) расположены все тексты пользователя, т. е. его оригинальные посты, комментарии, адресованные другим людям, и тексты при ретвитах. На странице «Медиа» (Media) сохраняются все медиафайлы пользователя (изображения и видео к твитам). Последний раздел «Нравится» (Likes) содержит все твиты, которым выбранный пользователь поставил соответствующую оценку – Like.

Таким образом, в социальной сети Twitter существует 2 глобальных объекта со своими полями: пользователь и пост. Каждый пользователь имеет следующие атрибуты: уникальный идентификатор, краткая информация о себе, список своих оригинальных постов, ретвитов и комментариев, список постов, которым пользователь поставил Like. А у каждого поста присутствуют: автор, текст, комментарии и списки пользователей, которые поставили отметку Like и / или ретвитнули этот пост. Учитывая весь функционал данной социальной сети, было выделено 4 вида взаимоотношений (взаимодействий) между пользователями:

- подписка друг на друга;
- лайки постов других пользователей;
- комментарии к постам других пользователей;

- ретвиты записей других пользователей.

На основе этих связей имеется возможность сформировать социальный граф пользователей. Социальный граф – граф, вершины которого представляют собой социальные объекты (ими могут быть аккаунты пользователей, сообщества), а ребра описывают отношения между ними. Причем, можно сделать как метаграф, содержащий ребра, соответствующие каждому из четырех взаимоотношений, так и отдельные графы для каждого вида взаимодействия.

Но наибольший интерес представляют взвешенные графы, вершины которых соответствуют пользователям, а ребра строятся в случае наличия хотя бы одного из взаимоотношений. При этом вес на ребрах в данном случае можно определять на основании данных об имевших место взаимодействиях по-разному, в зависимости от задач, поставленных перед исследователями сети скачанных объектов. Суммарный вес ребра может быть посчитан, например, как линейная комбинация весов, соответствующих каждому из зафиксированных взаимодействий пользователей. Эти отдельные веса для взаимоотношений и были приняты как параметры при построении взвешенных социальных графов.

Таким образом, определим (F, L, C, R)-модель информационного взаимодействия в сети Twitter как взвешенный граф $G(V, E)$, у которого на множестве ребер E весовая функция $w(\mathrm{e})$ с неотрицательными значениями задана следующим образом:

$$w(\mathrm{e}) = F \times \delta_e^F + L \times \delta_e^L + C \times \delta_e^C + R \times \delta_e^R,$$

где

$$\delta_e^F = \begin{cases} 2, & \text{если оба инцидентных пользователя подписаны друг на друга,} \\ 1, & \text{если хотя бы один из инцидентных пользователей подписан на другого,} \\ 0, & \text{иначе;} \end{cases}$$

$$\delta_e^L = \begin{cases} 2, & \text{если между инцидентными пользователями есть взаимые Like,} \\ 1, & \text{если есть хотя бы один Like между инцидентными пользователями,} \\ 0, & \text{иначе;} \end{cases}$$

$$\delta_e^C = \begin{cases} 2, & \text{если между инцидентными пользователями есть взаимые Reply,} \\ 1, & \text{если есть хотя бы один Reply между инцидентными пользователями} \\ 0, & \text{иначе;} \end{cases}$$

$$\delta_e^R = \begin{cases} 2, & \text{если между инцидентными пользователями есть взаимые Retweet,} \\ 1, & \text{если есть хотя бы один Retweet между инцидентными пользователями,} \\ 0, & \text{иначе.} \end{cases}$$

Под взаимностью тут понимаются действия от каждого из двух пользователей, но не обязательно к одному и тому же твиту. Так, например, (1, 3, 2, 4)-модель соответствует ситуации, когда вес подписок – 1, вес лайков – 3, вес комментариев – 2, вес ретвитов – 4.

Импорт данных

Входными данными для задачи импорта является список исходных постов, заданных оператором. Это могут быть обнаруженные ранее посты схожей или противоположной направленности об одном событии или группе событий, предположительные вбросы ложной / искаженной информации, рекламные сообщения и т. п. Никаких существенных ограничений на количество этих постов или их свойства не налагается.

Задача состоит в импорте данных об имевшем место взаимодействии пользователей социальной сети с этими постами и между собой. На основе этих данных на следующем шаге и будет построен социальный граф. А результат работы импорта представляет собой файл в специальном XML-подобном унифицированном формате AVS, который позволяет задать

вершины и ребра графа. Различные данные о вершинах и ребрах графа задаются в AVS-файле как атрибуты вершин и ребер соответственно. Важно отметить, что основные данные (уникальные имена пользователей-вершин и веса ребер) сохраняются в самом AVS-файле. В то же время тексты постов и комментариев могут суммарно быть достаточно большого объема, поэтому для каждого пользователя сохраняются в отдельном файле, ссылка на который находится в одном из атрибутов соответствующей вершины графа, описанной в AVS-файле. Далее такой файл может быть обработан сторонним программным обеспечением.

В ходе импорта данных можно выделить два основных этапа. На первом этапе для заданного списка исходных постов в зависимости от задач и настроенной конфигурации скачиваются списки пользователей, провзаимодействовавших с заданными постами. Это могут быть три списка пользователей:

- лайкнувшие данную запись;
- репостнувшие данную запись;
- прокомментировавшие данную запись.

На основе этой информации путем объединения пользователей из всех трех множеств формируется список вершин будущего социального графа. Более никакие вершины в граф не добавляются.

На втором этапе импортируются заданные согласно конфигурации взаимоотношения между пользователями. Как было сказано ранее, на основе данных из Твиттера можно построить различные взвешенные графы с учетом заданных параметров – весов каждого из 4 типов взаимодействий пользователей.

В итоге второго этапа работы формируются ребра графа, далее им присваиваются веса в зависимости от выбора модели, и сформированный граф записывается в AVS-файл.

Подходы к импорту данных из Twitter

Базовым методом для получения данных из Твиттера является использование API (Application Programming Interface) [1]. Но он имеет ряд ограничений: например, количество запросов на получение списка фолловеров пользователя лимитировано пятнадцатью запросами за пятнадцать минут, также сам функционал API ограничен, с его помощью нельзя скачать списки постов, которые лайкнул, ретвитнул или прокомментировал выбранный пользователь.

И главное – с сентября 2018 г. Твиттер ограничил выдачу новых аккаунтов для разработчиков. Платформа Твиттер для разработчиков отличается от обычных аккаунтов тем, что дополнительно предоставляется множество продуктов и инструментов API, которые позволяют автоматизировать многие процессы Твиттер в реальном времени. Например, можно настроить автоматический постинг твитов с других платформ, делать автоматические интеграции рекламы и совершать импорт данных.

Поэтому для импорта данных в финальной реализации приложения был выбран метод веб-скрапинга (получение данных путем извлечения их со страниц веб-ресурсов) [7], который также имеет свои ограничения. Так, при скачивании списков пользователей, лайкнувших или прокомментировавших выбранный пост, Твиттер ограничивает выдаваемые списки только 50–80 пользователями. Но, с другой стороны, данный метод позволяет импортировать всю информацию о пользователях (список читателей, ретвитов, лайков, комментариев), а ограничения на количество запросов в минуту не такие жесткие, как при использовании API. Здесь существенен тот факт, что спустя определенное время Твиттер начинает блокировать запросы с одного ip-адреса, поэтому приходится искусственно уменьшать частоту запросов, а также делать паузы в скачивании. Но, несмотря на это, метод веб-скрабинга позволяет скачивать информацию о взаимодействии между пользователями в более короткий срок.

[1] Twitter API. URL: https://developer.twitter.com/en/docs/basics/getting-started (дата обращения 15.02.2021).

Импорт вершин социального графа

Для импорта социального графа сначала выбирается, какая информация по всем указанным в исходных данных постам будет импортирована. Для оператора предоставлено три опции для множественного выбора: лайки, ретвиты и комментарии. При выборе этих элементов будут скачаны соответствующе списки пользователей. Иначе говоря, при выборе лайков, по каждому посту будет скачан список пользователей, которые лайкнули их, при выборе ретвитов будут импортированы списки ретвитнувших посты и при выборе комментариев соответственно формируются списки прокомментировавших.

Для импорта этих данных используется инструмент автоматизации действий веб-браузера – Selenium [2], а именно программная библиотека языка Python Selenium WebDriver [3] с использованием ChromeDriver [4] и пакет Python для анализа HTML Beautiful Soup [5]. Сначала с помощью веб-драйвера происходит аутентификация в Твиттере, затем для каждого поста посещаются и скачиваются HTML-коды следующих URL-страниц:

- при выборе лайков: https://twitter.com/author/status/postID/likes/;
- при выборе ретвитов: https://twitter.com/author/status/postID/retweets/;
- при выборе комментариев: https://twitter.com/author/status/postID/.

Дополнительно для страницы с комментариями перед скачиванием HTML-кода совершается прокрутка страницы до самого низа. Это необходимо, чтобы охватить весь список комментариев, которые изначально не помещаются при посещении страницы.

После этого скачанные HTML-коды страниц обрабатываются с помощью библиотеки Beautiful Soup. Для страниц каждого вида была проанализирована структура хранения данных и были определены нужные теги, по которым хранятся имена пользователей, совершивших действие. На этом этапе из HTML-кодов страниц для каждого поста формируются и сохраняются списки пользователей, лайкнувших, ретвитнувших и прокомментировавших посты (в зависимости от изначально выбранных флагов). Далее все списки пользователей объединяются в один users_list – список пользователей-вершин будущего социального графа.

Результатом данного этапа является формирование списка вершин. Первый этап производится единожды для заданных постов. Второй этап построения ребер может быть применен к списку вершин многократно, его итогом будет уже создание графа в зависимости от параметров.

Настройка импорта ребер социального графа

Далее для импорта ребер выбирается, какие виды отношений между пользователями будут использоваться для построения взвешенного графа. Есть четыре возможности: «Подписка», «Лайки», «Комментарии» и «Ретвиты», при выборе которых для каждого пользователя из предыдущего пункта (список users_list) будут импортированы списки его фолловеров, списки постов, которые он лайкает, комментирует, ретвитит соответственно. На основе этих списков далее будут построены ребра графа.

Также при выборе возможности любого типа, пользователю предоставляется выбор веса этого взаимодействия, что повлияет на веса ребер в итоговом графе.

Рассмотрим на примерах механизм работы данного этапа. Пусть выбран только пункт «Подписка» с весом F. Это означает, что если два пользователя не подписаны друг на друга,

[2] Инструмент автоматизации действий веб-браузера. URL: https://www.selenium.dev (дата обращения 15.02.2021).

[3] Библиотека Python Selenium. URL: https://selenium-python.readthedocs.io (дата обращения 15.02.2021).

[4] ChromeDriver. URL: https://sites.google.com/a/chromium.org/chromedriver (дата обращения 15.02.2021).

[5] Библиотека Python Beautiful Soup. URL: https://www.crummy.com/software/BeautifulSoup/ (дата обращения 15.02.2021).

то ребра между ними не будет; если присутствует односторонняя подписка, то вес ребра между вершинами будет F; а при взаимной подписке – $2F$.

При выборе флага «Лайки» с весом L смотрятся последние 50 отметок Like каждого пользователя. Если среди этого списка из 50 твитов присутствует хотя бы один лайк первого пользователя второму, то к весу ребра между ними прибавляется L, если аналогично у второго есть лайк к первому, то вес ребра становится $2L$. Учитывается только наличие или отсутствие лайков с каждой стороны, если количество лайков с одной стороны больше одного, то вес ребра все равно увеличивается только на L.

При активации флага «Комментарии» с весом C импортируются последние 50 текстов пользователя (его оригинальные посты и комментарии к другим). При наличии комментария к другому пользователю вес C, при взаимных – $2C$. Аналогично лайкам и комментариям учитывается только наличие или отсутствие комментариев с каждой стороны.

Аналогичная схема при выборе флага для «Ретвитов». С весом R скачиваются последние 50 постов пользователя включая ретвиты. Если среди них присутствует ретвит другого пользователя, то к весу ребра между ними прибавляется R, при взаимных репостах вес ребра увеличивается до $2R$. Так же как в лайках, учитывается только наличие или отсутствие ретвитов с обеих сторон, а количество больше одного не влияет на увеличение веса.

В итоге при формировании взвешенного графа все веса по каждому выбранному взаимодействию между двумя пользователями складываются, и получается итоговый вес ребра в графе. Разберем пример: пусть выбраны флаги «Подписка», «Лайки», «Комментарии» и «Ретвиты» с весами 1, 3, 2 и 4 соответственно. Два пользователя имеют взаимную подписку; первый ставит лайки второму, но второй не ставит первому; второй комментирует посты первого, но первый не комментирует посты второго; ретвиты они оба не делают. Тогда вес ребра между ними будет равен $(1 + 1) + 3 + 2 + 0 = 7$.

Описанные веса ребер строятся для каждой пары пользователей из списка вершин графа users_list, и на основе этих данных формируется итоговый взвешенный граф.

Также на данном этапе есть еще одна возможность – скачать последние 40 текстов (посты, комментарии, тексты к ретвитам) каждого пользователя, являющегося вершиной социального графа.

После выбора всех настроек начинает работать алгоритм импорта данных пользователей. В этом алгоритме также применяется Selenium WebDriver с использованием ChromeDriver и пакет Python Beautiful Soup. А также применяется библиотека Python Twint [6], основанная на методе скрапинга веб-страниц. Далее детально рассмотрим этапы его работы.

Импорт ребер социального графа

Для каждой выбранной опции запускается свой алгоритм. Соответствующий алгоритм применяется для всех пользователей из общего списка вершин users_list.

1. **Импорт подписок.** С помощью Selenium WebDriver и Python Beautiful Soup скачивается список читателей каждого пользователя. Все списки объединяются в единую структуру данных – словарь, где никам пользователей из users_list соответствуют списки их фолловеров.

2. **Импорт лайков.** Также с помощью Selenium WebDriver и Python Beautiful Soup для каждого пользователя импортируется список последних 50 постов, которые лайкает выбранный юзер, который впоследствии преобразуется в pandas DataFrame. Далее скачанные данные обрабатываются, и для каждого пользователя из users_list формируется два списка: список id постов и список никнеймов пользователей, которых пользователь отметил лайком. После этого полученные списки по всем пользователям также объединяются в словарь.

[6] Библиотека Python Twint. URL: https://github.com/twintproject/twint (дата обращения 15.02.2021).

3. **Импорт ретвитов.** Для данного пункта используется метод Profile библиотеки Python Twint, с помощью которой импортируется 50 оригинальных постов и ретвитов каждого пользователя с идентификатором автора поста. По этому идентификатору отсеиваются ретвиты от оригинальных постов пользователя и формируется список репостов. Далее, аналогично лайкам, с каждым пользователем в словаре сопоставляется список никнеймов и id постов, которые он ретвитил.

4. **Импорт комментариев и текстов.** В этом случае применяется метод Search, который возвращает оригинальные твиты, ответы и тексты к ретвитам пользователя в виде структуры pandas DataFrame, где есть следующие поля: id, conversation_id, reply_to, retweet, text. Поле retweet указывает, является ли данный объект ретвитом. Далее по conversation_id разделяем комментарии и оригинальные посты автора. Если пост является комментарием, то conversation_id равен id оригинального поста, к которому сделан данный комментарий, и reply_to – имя автора этого поста. А в случае оригинального поста conversation_id равно id, а поле reply_to пустое. Таким образом, из общей структуры можно выделить список постов и пользователей, которых комментирует автор, и составить словарь аналогичный лайкам, где каждому пользователю из users_list будет соответствовать два списка.

Более того, с помощью данного метода можно также скачать последние тексты пользователей, разделяя их по видовым группам (оригинальный пост, комментарий или текст ретвита).

Несомненным плюсом использования Python Twint при импортировании ретвитов и комментариев является тот факт, что импорт списков по каждому пользователю можно совершать в несколько потоков. Так, с помощью библиотеки Python Multiprocessing импорт работает в 4 потока.

После этого для каждого пользователя все скачанные ранее списки объединяются, и формируется единая структура данных pandas DataFrame. Затем все списки пользователей пересекаются со списком ников users_list. Так в структуре данных остается информация о лайках, ретвитах, комментариях только пользователей из списка вершин графа. Процесс пересечения происходит уже локально, после импорта данных, а не во время скачивания. Это позволяет сократить время работы, совершая меньшее количество запросов, а также сохранить все данные о пользователях для дальнейшего расширенного анализа. В итоге, после выполнения этой части алгоритма, на основе получившейся структуры строится финальный граф с учетом всех весов ребер.

В конце данного этапа после описанных выше процедур сам граф сохраняется в формате AVS. При необходимости можно выбрать другие варианты импорта взаимоотношений или изменить их веса. Скачанная до этого информация будет использована снова, а недостающая информация о пользователях будет импортирована. Другими словами, при смене только весов взаимодействий ничего дополнительного скачиваться не будет, будет только сформирован заново взвешенный граф, а при добавлении нового типа взаимодействия сначала импортируется дополнительная информация, после чего на основе старых и новых данных сформируется новый граф. Далее полученные графы импортируются в новые AVS-файлы.

Пример импорта данных – скачивание одного поста по (1, 3, 2, 4)-модели

В сети Twitter по состоянию на вечер 03.02.2021 был скачан следующий пост (от 02.02.2021), посвященный новости об успешном завершении третьей фазы клинических испытаний Sputnik V – вакцины от новой коронавирусной инфекции Covid-19: https://twitter.com/sputnikvaccine/status/1356580985127792650.

Исходный граф, построенный на основе импорта из Twitter по описанному в данной работе алгоритму в соответствии с (1, 3, 2, 4)-моделью, содержит 268 вершин и 100 ребер (рис. 1). Средний вес ребра равен 2,3, максимальная степень в графе – 12. При этом 174 вершины

изолированные – это пользователи, которые взаимодействовали с исходным постом, но никак не взаимодействовали с остальными скачанными пользователями (и те с ними – тоже).

После удаления изолированных вершин и применения к графу алгоритма Infomap [8] для разбиения на непересекающиеся сообщества получаем граф $\boldsymbol{G'}$. В нем выделено 15 сообществ (рис. 2). Три самых больших из них – $\boldsymbol{S_0}$, $\boldsymbol{S_1}$ и $\boldsymbol{S_2}$ содержат 41, 15 и 10 вершин соответственно. В каждом из этих сообществ, в свою очередь, аналогично может быть выделена структура (рис. 3).

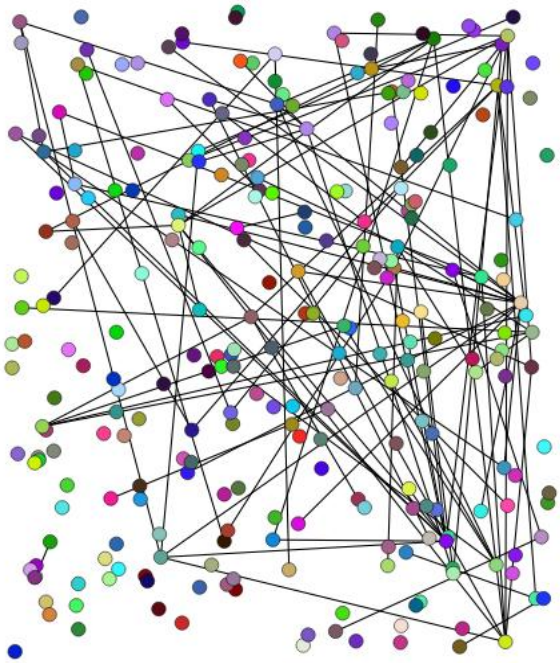


Рис. 1. Исходный граф **G**
Fig 1. The original graph **G**

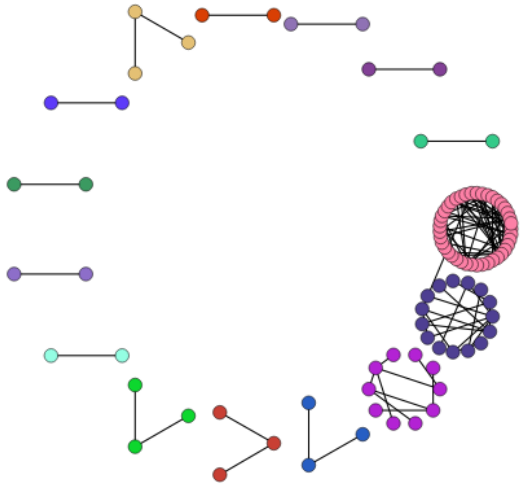
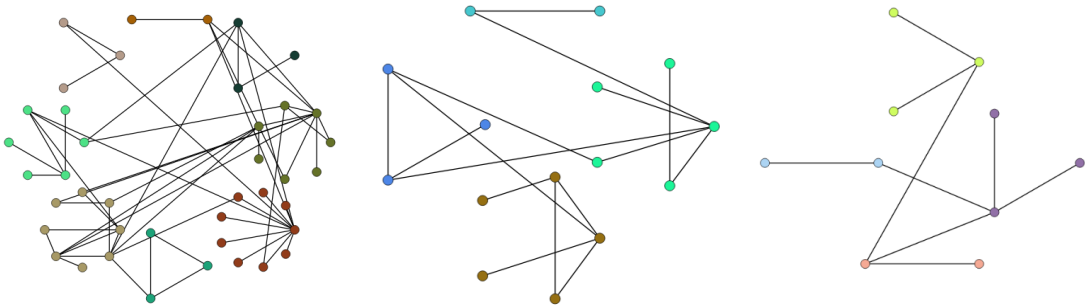


Рис. 2. Разбиение графа $\boldsymbol{G'}$ на 15 сообществ
Fig. 2. Splitting the graph $\boldsymbol{G'}$ into 15 communities



а б в

Рис. 3. Внутреннее разбиение сообществ: *а* – $\boldsymbol{S_0}$; *б* – $\boldsymbol{S_1}$; *в* – $\boldsymbol{S_2}$
Fig 3. Internal partition of the community: *a* – $\boldsymbol{S_0}$; *b* – $\boldsymbol{S_1}$; *c* – $\boldsymbol{S_2}$

Как видно, сообщество $\boldsymbol{S_2}$ не обладает выраженными вершинами-лидерами. Такая конструкция сообщества была названа в работе [9] «созвездием третьего рода». В сообществе $\boldsymbol{S_1}$ и особенно в $\boldsymbol{S_0}$ уже можно выделить лидеров мнений. Из четырех вершин с наибольшими степенями две получают их за счет всех взаимных действий (рис. 4), при этом не имея особых связей с остальными вершинами графа.

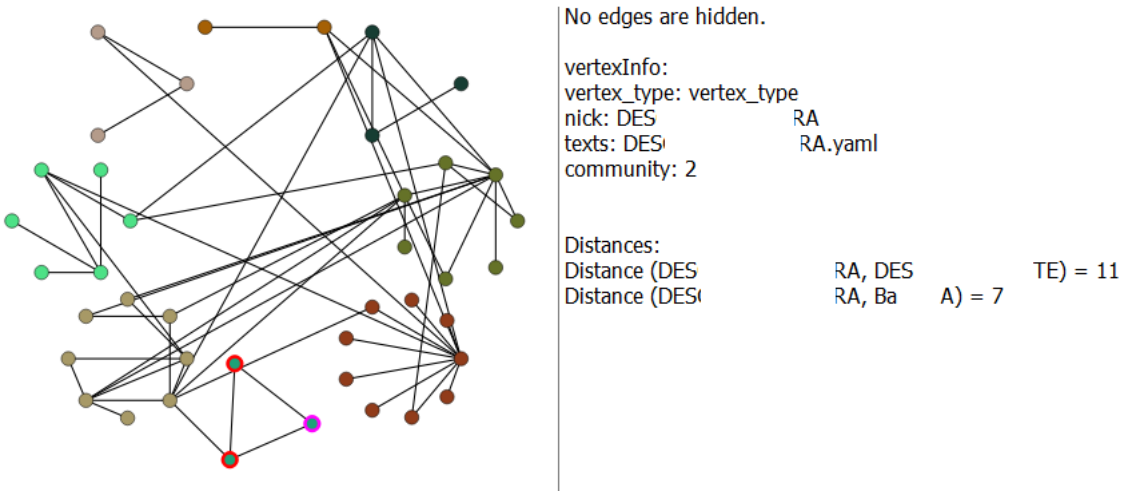


Рис. 4. Вершины с высокими степенями, но малым числом связей
Fig. 4. Vertices with high degrees but few connections

Представленный подход к импорту позволяет получить и исследовать таблицу со значениями по степеням и взвешенным степеням вершин. Посмотрим на вершины с высокими значениями этих показателей (табл. 1).

Две другие вершины в сообществе $\boldsymbol{S_0}$ являются явно выраженными лидерами мнений, на что указывают как высокая степень, так и связи с остальными вершинами (рис. 5). Таким образом, описанный алгоритм и $(1, 3, 2, 4)$-модель построения графов позволяют получить взвешенные графы, в которых можно выделять содержательные сообщества и искать лидеров мнений.

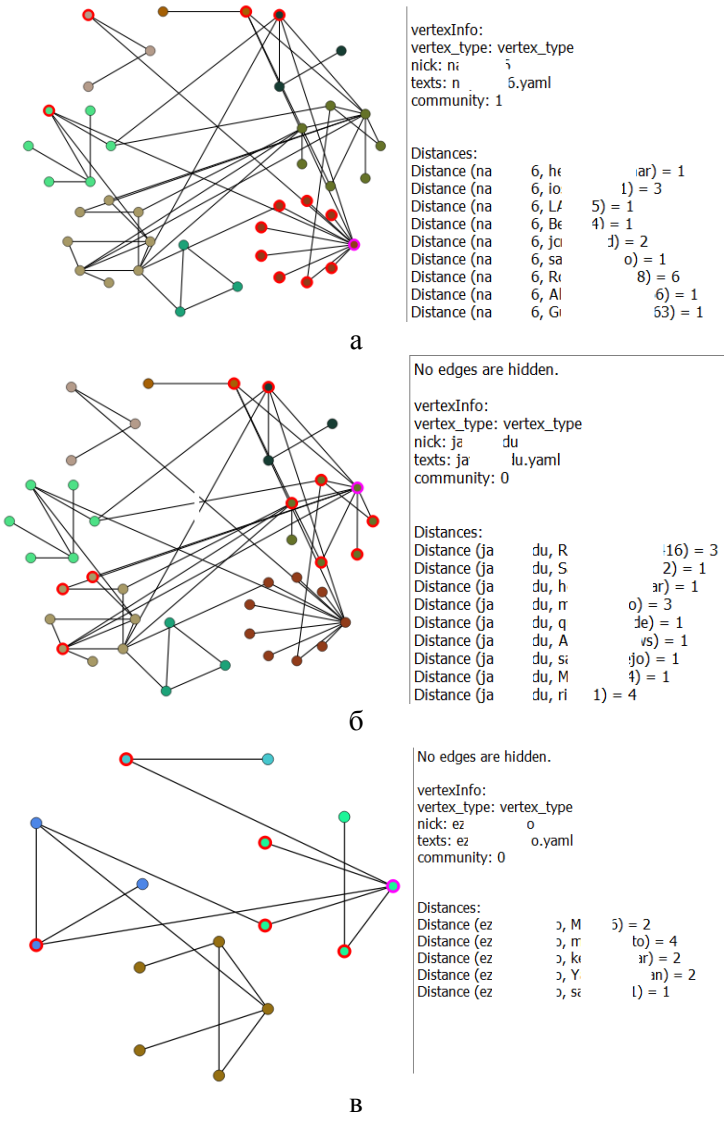
Таблица 1

Вершины графа G' с высокими значениями степеней

Table 1

Vertices of the graph G' with high degree values

Номер сообщества	Вершина	Рисунок	Степень вершины	Взвешенная степень вершины
0	na***6	5, *а*	12	25
0	DES***RA	4	2	18
0	DES***TE	4	3	17
0	ja***u	5, *б*	10	17
1	ez***o	5, *в*	5	11



а

б

в

Рис. 5. Вершина na***6 – лидер в S_0 (*а*);
вершина ja***u – лидер в S_0 (*б*); вершина ez***o – лидер в S_1 (*в*)

Fig 5. Vertex na***6 – leader in S_0 (*a*);
vertex ja***u – leader in S_0 (*b*); vertex ez***o – leader in S_1 (*c*)

Пример второй – скачивание нескольких постов и разные модели

В сети Twitter по состоянию на вечер 27.05.2020 были скачаны 8 актуальных постов, касающихся действий властей города Москвы в рамках борьбы с новой коронавирусной инфекцией Covid-19, а также относящиеся к ним комментарии, лайки, ретвиты. Среди постов были как официальные заявления руководства города в СМИ и своих аккаунтах, так и остросоциальные сообщения провокационного характера:

- https://twitter.com/rianru/status/1265550010000818180
- https://twitter.com/VRSoloviev/status/1265627361397166080
- https://twitter.com/rianru/status/1265630484488470528
- https://twitter.com/rianru/status/1265627223907860481
- https://twitter.com/tass_agency/status/1265628021538553861
- https://twitter.com/izvestia_ru/status/1265633807451009024
- https://twitter.com/moslenta/status/1265627960746352642
- https://twitter.com/meduzaproject/status/1265627619078389760
- https://twitter.com/EchoMskRu/status/1265627900373536768
- https://twitter.com/SobolLubov/status/1265629006478614529

Скачивание проходило с генерацией взвешенного графа в трех вариациях со следующими параметрами:

- (1, 3, 0, 0)-модель: без учета комментариев и ретвитов (вес подписок – 1, вес лайков – 3);
- (1, 3, 0, 4)-модель: без учета комментариев, но с учетом ретвитов (вес подписок – 1, вес лайков – 3, вес ретвитов – 4);
- (1, 3, 2, 4)-модель: с учетом комментариев и ретвитов (вес подписок – 1, вес лайков – 3, вес комментариев – 2, вес ретвитов – 4).

Также были скачаны и соответствующие тексты для дальнейшего анализа, ссылки на которые хранятся в AVS-файлах графов как длинные атрибуты. По итогам получено три соответствующих графа (табл. 2).

Таблица 2

Данные о скачанных графах

Table 2

Data about downloaded graphs

Граф	n	m	Mean edges weight	Mean vertex degree	Max vertex degree	Взвешенный диаметр
$\boldsymbol{G_1}$	632	845	2.257	1.337	92	23
$\boldsymbol{G_2}$	632	906	2.599	1.434	97	27
$\boldsymbol{G_3}$	632	1002	2.767	1.585	126	28

Далее для каждого из графов был применен алгоритм Infomap [8] для выделения неявных сообществ (табл. 3, 4). Необходимо отметить, что в полученных графах в соответствии с алгоритмом скачивания имеется большое количество висячих вершин с ребрами единичного веса. Данные вершины соответствуют пользователям, имевшим взаимодействие с исходными постами, но при этом с остальными вершинами связанные лишь фолловерством.

В результате анализа графа сторонним программным обеспечением получены лидеры мнений небольших сообществ, участвовавших в обсуждении и распространении исходных постов. Причем в ряде случаев в зависимости от графа сообщества формируются по-разному, что подчеркивает полезность скачивания по разным конфигурациям и использование соответствующих моделей.

Таблица 3

Данные о разбиении графов

Table 3

Data on graph partitioning

Граф	Число выделенных сообществ	Число сообществ			
		более 3вершин	3 вершины	2 вершины	1 вершина
G_1	255	26	6	6	217
G_2	242	34	7	3	198
G_3	216	33	4	6	173

Таблица 4

Данные о разбиении графов

Table 4

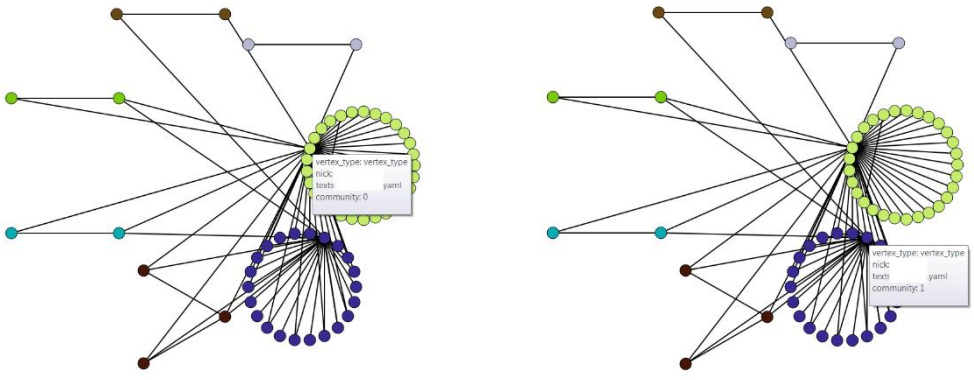
Data on graph partitioning

Граф	Размеры топ-5 сообществ	Средний размер		
		сообщества	сообщества без учета единичных	топ-10
G_1	62-39-32-30-25	2,47	10,92	26,9
G_2	40-37-36-31-30	2,61	9,86	26,7
G_3	49-44-44-33-25	2,92	10,67	29,3

Посмотрим на следующие примеры.

Пример 1. На графе G_1 самое большое сообщество состоит из 62 вершин, причем внутри имеется явное разделение (рис. 6).

Для графа G_2 те же вершины разбились на сообщества иначе – разделившись на два разных за счет большего веса на ребрах между ними (рис. 7).



а б

Рис. 6. Один из лидеров сообщества, состоящего из двух центральных и пяти близких к ним (*а*); еще один лидер того же сообщества (*б*)

Fig. 6. One of the leaders of a community consisting of two central and five close to them (*a*); another of the leaders of the same community (*b*)

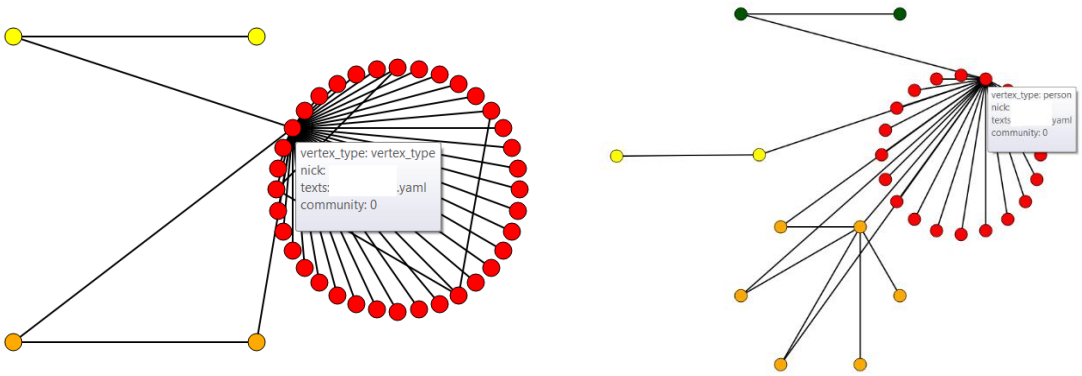
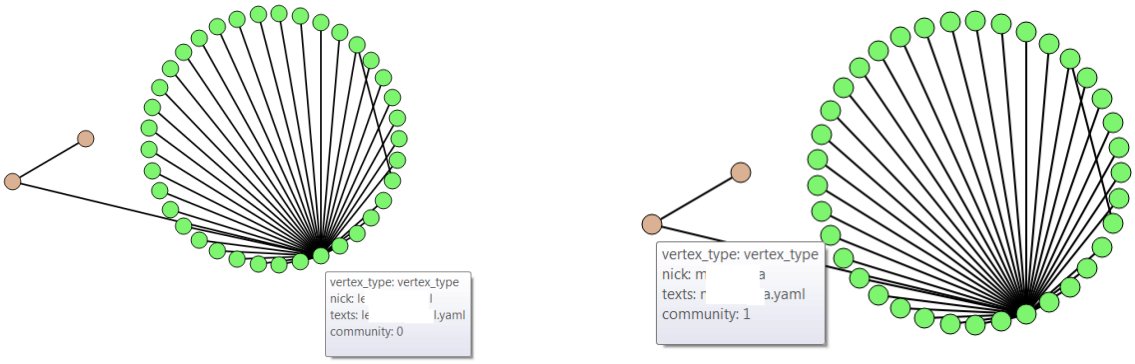
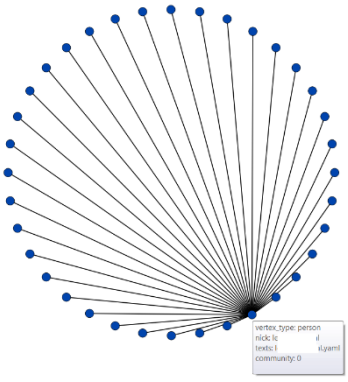


Рис. 7. Первый из лидеров теперь в своем сообществе (*а*); второй из лидеров также в своем сообществе (*б*)
Fig. 7. The first of the leaders is now in his community (*a*); the second of the leaders is also in his community (*b*)

Пример 2 состоит в наличии «хвоста» у одного из сообществ для графа $\boldsymbol{G_1}$ и его отсутствия для $\boldsymbol{G_2}$ (рис. 8).



а б



в

Рис. 8. Лидер сообщества @le***al (офиц. СМИ), граф $\boldsymbol{G_2}$ (*а*);
наличие «хвоста» в сообществе @le***al в графе $\boldsymbol{G_2}$ (*б*);
сообщество @le***al в графе $\boldsymbol{G_2}$ уже без «хвоста» (*в*)
Fig. 8. Community Leader @le** * al (ofic. Media), graph $\boldsymbol{G_2}$ (*a*);
the presence of a “tail” in the @le***al community in the $\boldsymbol{G_2}$ column (*b*);
the @le***al community in the $\boldsymbol{G_2}$ column is already without a “tail” (*c*)

Пример 3 состоит в рассмотрении вершины, соответствующей пользователю с ником ***@Mr***ay***. В графе G_1 у этой вершины 47 смежных, суммарная степень равна 73. При этом в графе G_2 степень вершины растет в 1,25 раза, а ее взвешенная степень растет почти в 2 раза (табл. 5). Это связано с большим числом ретвитов данного пользователя.

Таблица 5

Данные о вершине @Mr***ay

Table 5

Data on the vertex @Mr***ay

Граф	Степень		Взвешенная степень	
	вершины	вершины в своем сообществе	вершины	вершины в своем сообществе
G_1	47	19	73	43
G_2	59	29	145	100
G_3	65	39	171	128

Например, исследуемая вершина теперь имеет общее ребро с весом 4 с вершиной, соответствующей пользователю ***@ib***hP***. Его появление обусловлено одним ретвитом. Но общее перестроение графа повлекли и обратные примеры, часть пользователей ушла в другие сообщества, например, в графе G_1 вершина ***@y***an*** и ее небольшое сообщество из 4 вершин были в общем сообществе с ***@Mr***ay***, а в графе G_2 – уже нет. При этом вес вершины ***@Mr***ay*** значительно вырос за счет ретвитов (9, *а*, *б*). Для графа G_3, в котором добавлены еще и комментарии, рост взвешенной степени вершины уже не такой резкий, но продолжается, тут вершина ***@y***an*** и ее сообщество возвращаются (рис. 9, *в*).

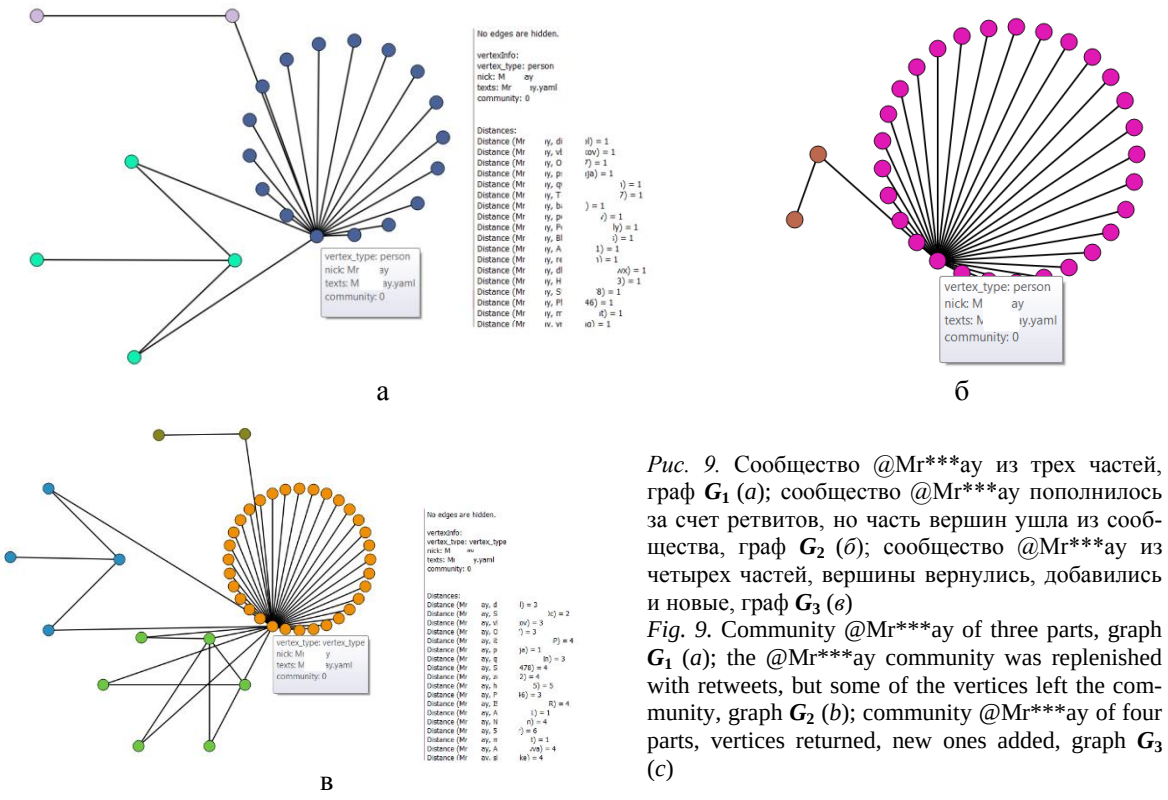


Рис. 9. Сообщество @Mr***ay из трех частей, граф G_1 (*а*); сообщество @Mr***ay пополнилось за счет ретвитов, но часть вершин ушла из сообщества, граф G_2 (*б*); сообщество @Mr***ay из четырех частей, вершины вернулись, добавились и новые, граф G_3 (*в*)

Fig. 9. Community @Mr***ay of three parts, graph G_1 (*a*); the @Mr***ay community was replenished with retweets, but some of the vertices left the community, graph G_2 (*b*); community @Mr***ay of four parts, vertices returned, new ones added, graph G_3 (*c*)

Корректируя конфигурацию для скачивания вершин, можно добиться графов с разным содержанием в зависимости от поставленной оператором цели. Первую конфигурацию – (1, 3, 0, 0)-модель – мы называем **графом общего сходства пользователей**. Третью конфигурацию – (1, 3, 2, 4)-модель – мы называем **графом информационного взаимодействия пользователей**. Таким образом, построены подходы к импорту данных и анализу катализаторов информационного воздействия в сети Twitter.

Заключение

В рамках данной работы описан выработанный авторами комплексный анализ возможностей и ограничений по импорту данных из социальной сети Twitter в контексте взаимодействия пользователей. В работе представлена методика построения (*F*, *L*, *C*, *R*)-моделей информационного взаимодействия, представлены примеры ее использования для выявления лидеров мнения групп пользователей и соответствующих катализаторов.

Авторами разработаны алгоритм и структуры данных для него, позволяющие построить взвешенные социальные графы на основе данных о взаимодействии пользователей Twitter. Описано реализованное программное обеспечение для импорта данных пользователей на основе заданных изначально оператором твитов-источников и конфигурационных параметров модели информационного взаимодействия.

Приведенные примеры показывают высокую степень актуальности достигнутых результатов. Авторам представляется целесообразным дальнейшее развитие методики построения моделей информационного взаимодействия и ее расширение на другие социальные сети.

Список литературы

1. **Лещёв Д. А., Сучков Д. В., Хайкова С. П., Чеповский А. А.** Алгоритмы выделения групп общения // Вопросы кибербезопасности. 2019. Т. 32, № 4. С. 61–71.
2. **Соколова Т. В., Чеповский А. А.** Анализ профилей сообществ социальных сетей // Системы высокой доступности. 2018. Т. 14, № 3. С. 82–86.
3. **Коломейченко М. И., Поляков И. В., Чеповский А. А., Чеповский А. М.** Выделение сообществ в графе взаимодействующих объектов // Фундаментальная и прикладная математика. 2016. Т. 21, № 3. С. 131–139.
4. **Roth M., Ben-David A., Deutscher D.** Suggesting Friends Using the Implicit Social Graph. In: KDD'10, July 25–28, 2010. Washington, DC, USA, 2010.
5. **Girvan M., Newman M.** Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 2002, vol. 99, no. 12, p. 7821–7826.
6. **Blondel V. D., Guillaume J. L., Lambiotte R., Lefebvre E.** Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008, no. 10, P10008. 12 p.
7. **Mitchell R.** Web Scraping with Python. Sebastopol, O'Reilly Media, 2015.
8. **Rosvall M.**, **Axelsson D., Bergstrom C. T.** The map equation. *The European Physical Journal Special Topics*, 2009.
9. **Chepovskiy A. A., Leshchev D. A., Khaykova S. P.** Core Method for Community Detection. In: Complex Networks & Their Applications IX. Vol. 1: Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020. Springer, 2021, p. 38–50. DOI 10.1007/978-3-030-65347-7_4

References

1. **Leschyov D. A., Suchkov D. V., Khaykova S. P., Chepovskiy A. A.** Algorithms to reveal communication groups. *Voprosy kiberbezopasnosti*, 2019, vol. 32 (4), p. 61–71. (in Russ.) DOI 10.21681/2311-3456-2019-4-61-71

2. **Sokolova T. V., Chepovskiy A. A.** Analiz profilej soobshchestv social'nykh setej. *Sistemy vysokoj dostupnosti*, 2018, vol. 14, no. 3, p. 82–86. (in Russ.)
3. **Kolomejchenko M. I., Polyakov I. V., Chepovskiy A. A., Chepovskiy A. M.** Vydelenie soobshchestv v grafe vzaimodejstvuyushchikh ob'ektov. *Fundamental'naya i prikladnaya matematika*, 2016, vol. 21, no. 3, p. 131–139. (in Russ.)
4. **Roth M., Ben-David A., Deutscher D.** Suggesting Friends Using the Implicit Social Graph. In: KDD'10, July 25–28, 2010, Washington, DC, USA, 2010.
5. **Girvan M., Newman M.** Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 2002, vol. 99, no. 12, p. 7821–7826.
6. **Blondel V. D., Guillaume J. L., Lambiotte R., Lefebvre E.** Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008, no. 10, P10008. 12 p.
7. **Mitchell R.** Web Scraping with Python. Sebastopol, O'Reilly Media, 2015.
8. **Rosvall M.**, **Axelsson D., Bergstrom C. T.** The map equation. *The European Physical Journal Special Topics*, 2009.
9. **Chepovskiy A. A., Leshchev D. A., Khaykova S. P.** Core Method for Community Detection. In: Complex Networks & Their Applications IX. Vol. 1: Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020. Springer, 2021, p. 38–50. DOI 10.1007/978-3-030-65347-7_4

Материал поступил в редколлегию
Received
29.03.2021

Сведения об авторах

Попов Владимир Александрович, студент магистратуры, Национальный исследовательский университет «Высшая школа экономики» (Москва, Россия)

vapopov_1@edu.hse.ru

Чеповский Александр Андреевич, кандидат физико-математических наук, доцент, Национальный исследовательский университет «Высшая школа экономики» (Москва, Россия)

aachepovsky@hse.ru

Information about the Authors

Vladimir A. Popov, Master's Student, National Research University Higher School of Economics (Moscow, Russian Federation)

vapopov_1@edu.hse.ru

Alexander A. Chepovskiy, PhD (Mathematics), Associate Professor, National Research University Higher School of Economics (Moscow, Russian Federation)

aachepovsky@hse.ru

УДК 004.8
DOI 10.25205/1818-7900-2021-19-2-92-101

Предсказание лавинной опасности методами машинного обучения

Н. А. Радеев

Новосибирский государственный университет
Новосибирск, Россия

Аннотация

На возникновение снежных лавин главным образом влияют метеорологические условия и конфигурация слоев снежного покрова. Методы машинного обучения имеют предиктивную силу и при должном качестве обучения способны предсказывать новые явления. При обучении моделей машинного обучения на данных о сходах лавин, метеорологических условиях и состоянии снежного покрова получен ансамбль моделей, предсказывающий возможность схода лавины. Представленная в статье модель использует для обучения данные о сходах лавин и метеорологические данные, что позволяет применять полученное решение в большем количестве горных районов, чем решения, использующие более широкий спектр менее доступных данных.

Данные о состоянии снежного покрова были сгенерированы программным пакетом SNOWPACK.

Ключевые слова

предсказание снежных лавин, машинное обучение, ансамбль, машина опорных векторов, snowpack, наивный байесовский классификатор, многослойный перцептрон, случайный лес, логистическая регрессия

Для цитирования

Радеев Н. А. Предсказание лавинной опасности методами машинного обучения // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 92–101. DOI 10.25205/1818-7900-2021-19-2-92-101

Avalanches Forecasting Using Machine Learning Methods

N. A. Radeev

Novosibirsk state university
Novosibirsk, Russian Federation

Abstract

The occurrence of snow avalanches is mainly influenced by meteorological conditions and the configuration of snow cover layers. Machine learning methods have predictive power and are capable of predicting new events. From the trained machine learning models, an ensemble is obtained that predicts the possibility of avalanches. The model obtained in the article uses avalanche data, meteorological data and generated data on the state of snow cover for training. This allows the resulting solution to be used in more mountainous areas than solutions using a wider range of less available data.

Snow data is generated by the SNOWPACK software package.

Keywords

snow avalanches forecasting, machine learning, ensemble, support vector machine, snowpack, naive bayes, perceptron, random forest, logistic regression

For citation

Radeev N. A. Avalanches Forecasting Using Machine Learning Methods. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 92–101. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-92-101

Введение

Оценка лавинной опасности является сложной задачей, для решения которой нужно специальное образование и опыт. Также данный процесс требует личного присутствия специалиста в месте, для которого производится оценка. А количество тестов, которые необходимо провести над снежным покровом для получения оценки, приближается к сорока. Всё это делает задачу оценки лавинной опасности очень трудозатратной.

Знание актуального значения оценки лавинной опасности может помочь предпринять превентивные меры по обеспечению защиты благосостояния инфраструктуры, здоровья и жизни людей. В настоящее время системы автоматической оценки лавинной опасности находятся в зачаточном состоянии, и нет решений, пригодных для применения. Таким образом, разработка алгоритма оценки лавинной опасности является актуальной задачей, так как количество специалистов, способных оценивать лавинную опасность, на порядки меньше, чем количество горных районов, в которых необходимо производить указанный вид оценки.

В данной статье описывается разработка хорошо масштабируемого алгоритма предсказания снежных лавин. Последовательно производится обзор существующих решений, описывается, как происходил поиск данных, их очистка, подготовка и предварительный анализ, а также разработка алгоритма оценки лавинной опасности с использованием методов машинного обучения. Обучение и проверка алгоритма происходит на данных из открытых источников, собиравшихся в течение двадцати лет. В их числе метеорологические наблюдения и сгенерированные данные о состоянии снежного покрова.

Обзор существующих работ

Есть множество работ, в которых лавины изучаются с помощью статистических методов, в числе которых и методы машинного обучения. Так, например, в работе [1] строится карта, на которой территории классифицируются как слабо, средне и сильно лавиноопасные на основании статистики о местах схода лавин, их количестве и физических параметров лавин. Было выяснено, что индекс топографической позиции, уклон, осадки и индекс влажности были наиболее полезными данными для обучения алгоритмов. В результате работы получена карта, на основе которой был сделан вывод, что в основном зона лавинной опасности располагается рядом с ручьями и горными склонами возле водных путей.

Все данные собирались из разных источников. В итоге собрано большое разнообразие данных, из которых были получены 14 переменных, используемых в алгоритмах. Данные разбивались в отношении 7 : 3 (7 частей на обучение, 3 на валидацию). Обучение происходило на двух алгоритмах: support vector machine (SVM) и multivariate discriminant analysis (MDA). Выбор объясняется тем, что SVM отлично обобщает и находит паттерны в данных, а MDA предназначен для работы с независимыми переменными, из которых он строит линейные комбинации так, чтобы события из одного класса были на небольшом расстоянии друг от друга.

В работе [2] обучен алгоритм для оценки лавинной опасности по метеорологическим данным, состоянию снежного покрова и другим данным, предоставленным лавинным центром. Модель представляла собой SVM. Данные представляли собой сведения о погоде за три дня и информацию о снежном покрове. Также в результате диалога с лавинным центром был получен список важных индикаторов лавин – около десяти величин, среди которых градиент температуры снега, изменение температуры в предыдущие два дня (любое резкое изменение температуры повышает лавиноопасность) и д. В итоге получилось 44 входных параметра.

Было отобрано 20 наиболее важных параметров. Среди них первые места занимают параметры, которые дал лавинный центр. В свою очередь, некоторые параметры из общедоступного прогноза оказались совсем бесполезными. В общей сложности тренировочные данные

покрывают период в 16 лет – с 1991 по 2007 г. (работа 2008 г., т. е. это максимально свежие данные на тот момент).

В результате работы был получен алгоритм, имеющий метрику точности около 86 % (т. е. среди всех ответов алгоритма 86 % совпали с реальными данными). При этом 28 % прогнозов о возможном сходе лавины (т. е. алгоритм предсказал лавину, но в жизни ее не было) и 8 % прогнозов об отсутствии лавины (алгоритм предсказал отсутствие, а в жизни лавина случилась) являются ложными.

Также существует множество работ, в которых так или иначе проводятся исследования снежного покрова, что имеет косвенное, но очень важное отношение к лавинам. Например, в работе [3] проводится подробный обзор типов снега, апробируется программа SNOWPACK для моделирования состояния снежного покрова, проводится верификация полученных результатов с помощью сравнения с реальными данными, собираемыми в горах Кавказа. А в работе [4] показывается, что использование сгенерированных данных о снежном покрове улучшает предсказания оценки лавинной опасности по пятибалльной шкале.

Развивается направление предсказания лавин с помощью специальных датчиков, устанавливаемых на лавинных склонах [5]. Датчики собирают статистику о колебаниях, возникающих в районе установки. Затем с помощью алгоритмов машинного обучения выделяют те колебания, которые соответствуют сходу лавин, а, например, не пролетевшему мимо самолету.

Обработка данных

Сбор данных

Так как полученное решение должно было эффективно масштабироваться на другие горные районы, то в работе желательно было использование данных, сбор которых не представляет большой сложности и легко автоматизируется. К сожалению, стоит отметить, что сбор статистики по сходу лавин трудоемок. Для этого нужно присутствие специалиста в горном районе, который ежедневно проводит ревизию склонов и фиксирует наличие следов свежих сходов лавин. Тем не менее, стоит ожидать в ближайшем будущем повсеместного появления автоматических детекторов лавин, которые достаточно будет установить в лавинных очагах и периодически собирать с них данные и оказывать техническую поддержку.

Были произведены поиски данных для одного горного района, которые содержали бы статистику схода лавин по дням и еще какие бы то ни было данные о районе, собираемые ежедневно. К счастью, метеорологические станции достаточно широко распространены и выкладывают свои данные в открытый доступ.

В ходе поисков была найдена база данных об окружающей среде Швейцарии. Из этой базы были получены данные о сходах лавин примерно за 20 лет, собираемые практически ежедневно, метеорологические измерения за тот же срок и данные о состоянии снежного покрова, собираемые специалистом примерно раз в две недели. Важным фактором было то, что все три набора данных были собраны в одном горном районе, что гарантировало их совместимость.

Предварительный анализ данных показал, что сведения о снежном покрове собирались недостаточно строго, плохо поддаются манипуляциям ввиду неудачного структурирования измерений и собирались слишком редко. Поэтому они оказались непригодными для использования в решении задачи данной работы.

В результате дальнейших изысканий было решено применить программное генерирование состояния снежного покрова на основе метеорологических измерений, которые собраны в достаточном количестве и были подходящего качестве. Для моделирования состояния снежного покрова использован программный пакет SNOWPACK, разработанный швейцарскими исследователями [6]. Данная программа имеет целый ряд подтверждений корректно-

сти своей работы, состоящий из сравнений результатов генерации с данными из реального мира для самых разнообразных мест земного шара.

Очистка и подготовка данных

Так как данные собирались людьми в реальном мире, то в них содержались дефекты вроде пропущенных значений, полей, заполненных не по правилам, и т. п.

Набор сведений о лавинах содержал множество данных: причина схода лавины, тип снега в лавине, размер лавины. В данной работе важен был лишь факт схода либо его отсутствие в течение дня. Поэтому на основе данных был получен список пар (дата, была ли лавина – да / нет).

Метеорологические данные имели пару недостатков. Так как данные начали собирать еще в 1990-е гг., то часть измерений тогда не проводили. В результате не оказалось записей по некоторым показателям за несколько лет. К счастью, эти данные не были нужны для генерации снежного покрова, поэтому от них можно было просто отказаться. Второй проблемой было измерение толщины снега, которое проводилось не слишком часто – раз в две недели. Интуиция подсказывала, что динамика изменения толщины снежного покрова должна играть роль в определении лавинной опасности, поэтому было принято решение разработать алгоритм моделирования данной величины.

Было принято упрощение – за срок в две недели толщина снега либо монотонно убывает, либо монотонно возрастает. В метеорологических данных указывалось количество осадков, выпавших в течение дня. Было решено общий прирост (убыль) толщины H снежного покрова за две недели поделить на общее число осадков P, выпавших в данный срок, – пусть это величина $s = H / P$. Затем недостающие записи восстанавливались пропорционально количеству осадков, выпавших в соответствующий день: если в день k выпало p осадков, то толщина снега в день k увеличилась на $s * p$. Для случая, когда толщина снега за 2 недели убывала, скорость убывания была обратно пропорциональна числу осадков, выпавших в соответствующий день. Таким образом, все недостающие значения были получены.

С данными о состоянии снежного покрова не было никаких проблем, потому что они были получены с помощью генератора. Вследствие этого они сразу были готовы к использованию.

Выделение значимых признаков

Для обучения алгоритма предсказания факта лавин имеющиеся данные нужно было преобразовать к виду (X, y), где X – вектор фиксированной длины, содержащий данные, описывающие точку в пространстве признаков, а y – целевая бинарная величина, которую алгоритм должен предсказать, получив на вход X.

Метеорологические данные имеют период 30 минут. В свою очередь, статистика по лавинам имеет период 1 день. Другими словами, множество метеорологических данных за сутки нуждалось в агрегации в небольшое количество величин. Например, из температуры были выделены три величины: максимальная температура за сутки, минимальная температура за сутки, разность максимальной и минимальной температуры за сутки. Таким образом, 24 числа были заменены тремя. Подобным образом, исходя из физического смысла показаний и интуитивной оценки важности того или иного показания для лавинной опасности, были выделены 23 переменные.

Данные о снежном покрове представляют собой численное описание свойств слоев снега: порядковый номер, толщина, высота над уровнем земли, тип снега и другие характеристики. Возникла проблема того, что количество слоев снега – это переменная величина. В начале зимы снега нет совсем – 0 слоев. А в январе количество слоев достигает нескольких сотен. Извлечь полезную информацию из такого массива данных не просто, учитывая, что в целом набор обучающих данных достаточно небольшой. Поэтому из данных о слоях снега были

взяты 11 величин, которые описывают снежный покров в целом, а не каждый слой индивидуально.

Таким образом, было получен вектор X длиной 34, состоящий из вещественных чисел. Все переменные были нормированы до диапазона [–1; 1]. Всего в обучающей выборке около 3 000 записей, из них 2 292 об отсутствии лавин и 701 о наличии.

Анализ данных

С помощью статистических функций и библиотек для визуализации данных произведен анализ. Была составлена матрица корреляций признаков, и из пар признаков, имеющих корреляцию больше 0,8, убран один из признаков. Данное действие значительно уменьшает размерность пространства признаков, при этом в данных остается признак, который хранит похожую в смысле корреляции информацию. В результате было удалено 6 признаков, осталось 28.

Далее для каждого признака данные были разбиты по целевой переменной на два множества, и для каждого множества была построена диаграмма размаха (ящик с усами). Таким образом, для каждого признака получились две диаграммы, сообщающие, как распределены данные, где их среднее, максимальное и минимальное значения, выбросы. В результате из обучающей выборки были удалены те признаки, у которых диаграммы размаха для обоих состояний (есть лавина / нет лавины) совпадали. У оставшихся признаков диаграммы заметно разнились. Это говорит о том, что оставшиеся признаки имеют отношение к факту схода лавины. В итоге было удалено 12 признаков, осталось 16.

Разработка алгоритма предсказания

Построение модели

Данные были перемешаны и разбиты на две выборки в отношении 9 : 1. Самая большая выборка нужна для обучения алгоритма (обучающая выборка). Оставшаяся часть данных разбивается на валидационную и тестовую выборки. С помощью валидационной выборки можно подбирать гиперпараметры алгоритма, а на тестовой выборке будут получены итоговые метрики качества.

Были обучены различные алгоритмы классификации: дерево решений, случайный лес деревьев решений, градиентный бустинг, машина опорных векторов, логистическая регрессия, наивный байесовский классификатор, многослойный перцептрон, k ближайших соседей. Каждый алгоритм имеет определенный набор параметров, регулируя которые, можно улучшить или ухудшить метрики качества, полученные на валидационной выборке. Добившись экспериментальным путем наилучших показателей на проверочных и тренировочных данных, алгоритмы были объединены в ансамбль по принципу бэггинга с равными весами. Иначе говоря, каждый алгоритм в ансамбле делал независимое предсказание, а затем итоговое предсказание ансамбля выбиралось простым голосованием по принципу большинства. Так как каждый алгоритм сам по себе имеет некие недостатки, то имеет смысл совместить результаты их работы и выбирать ответ путем голосования среди алгоритмов. Например, один алгоритм хорошо распознавал записи, в которых происходят лавины, но при этом причислял к данному классу и записи, которые на самом деле относятся к классу «нет лавины». А другой алгоритм поступал наоборот. Тогда, усреднив результаты работы обоих алгоритмов, можно было ожидать, что вместе они будут давать более приближенную к правде оценку.

При обучении и проверке моделей машинного обучения использовался ряд метрик: точность, полнота, аккуратность, f1-мера. При этом важно было создать алгоритм, который как можно реже давал бы ложный положительный ответ, – когда в жизни лавина случилась, а алгоритм «сказал», что лавины быть не должно. Таким образом, даже если все остальные метрики были слегка выше, выбирался тот вариант, у которого полнота класса «лавина есть» была больше.

Список испробованных алгоритмов и их лучшие результаты на валидационной выборке приведены в табл. 1.

Таблица 1

Сравнение алгоритмов

Table 1

Comparison of algorithms

№ п/п	Алгоритм	TP	TN	FP	FN	ACC	F1
1	Дерево решений	30	60	26	27	0.63	0.53
2	Градиентный бустинг	31	80	14	25	0.74	0.61
3	Случайный лес деревьев решений	27	89	5	29	0.77	0.61
4	*K*-ближайших соседей	21	83	11	35	0.69	0.48
5	Машина опорных векторов с полиномиальным ядром 3-го порядка	35	52	59	4	0.58	0.53
6	Машина опорных векторов с линейным ядром	48	54	40	8	0.68	0.67
7	Машина опорных векторов с сигмоидным ядром	59	27	25	29	0.61	0.69
8	Машина опорных векторов с радиально-базисной функцией в качестве ядра	47	54	40	9	0.67	0.66
9	Классификатор на линейном дискриминантном анализе	27	80	14	29	0.71	0.56
10	Наивный байесовский классификатор	42	59	35	14	0.67	0.63
11	Многослойный перцептроп	38	73	21	18	0.74	0.66
12	Логистическая регрессия	50	56	38	6	0.71	0.69

Примечание: TP (True Positive) – лавина была и предсказана; TN (True Negative) – лавины не было, и это предсказано; FP (False Positive) – лавины не было, но она была предсказана; FN (False Negative) – лавина была, но было предсказано, что нет.

В результате комбинирования моделей в ансамбли показатели итогового алгоритма были заметно улучшены. В основе ансамбля легли три версии машины опорных векторов, которые очень редко предсказывали отсутствие лавины для записи, когда на самом деле лавина была. Но в обратную сторону они справлялись не так хорошо и часто предсказывали лавину, когда ее не было. Чтобы компенсировать данный недостаток, в ансамбль были добавлены модели с обратной балансировкой результатов либо модели, в равной степени ошибавшиеся в обоих случаях.

При построении ансамбля основной идей было взять наиболее «безопасные» модели, т. е. такие, которые меньше всего склонны причислять события из класса «сошла лавина» к классу «лавины не было», и дополнить ансамбль теми моделями, которые могли бы компенсировать недостатки уже взятых в ансамбль и уменьшить ошибку в определении класса «лавины не было».

Был проведен ряд экспериментов по комбинированию моделей, которые хорошо дополняли друг друга (табл. 2). Сравнение производилось на валидационной выборке.

Из полученных ансамблей в первую очередь выбирались те, у которых FN наименьший. Ранее говорилось, что это самая важная метрика, так как предсказать отсутствие лавины тогда, когда она была на самом деле, опасно для жизни людей, и таких случаев следует избегать. Наименьший FN у ансамблей под номерами 2, 5, 6–9. При этом сразу можно отсеять

номера 5–7, так как у них FP больше при тех же значениях FN (рис. 1). Таким образом, остались ансамбли 2, 8, 9. Из них выбор был сделан в пользу ансамбля с наименьшим числом ошибок, т. е. под номером 8. Ансамбль номер 2 хоть и является более «безопасным» в плане предсказаний, метрики качества имеет значительно более плохие. В свою очередь, ансамбль номер 9 дает примерно те же результаты, что и 8. Но так как на тестовой выборке у обоих ансамблей FN одинаковый, а FP меньше у номера 8, то решено остановить выбор на ансамбле номер 8.

Таблица 2

Сравнение ансамблей

Table 2

Comparison of ensembles

№ п/п	Номера моделей, включенных в ансамбль	TP	TN	FP	FN	ACC	F1
1	3, 12	26	89	5	30	0.77	0.6
2	3, 5, 12	50	57	37	6	0.71	0.7
3	3, 5, 11, 12	43	75	19	13	0.79	0.73
4	3, 5, 10, 12	44	70	24	12	0.76	0.71
5	3, 5, 10, 11, 12	49	62	32	7	0.74	0.72
6	3, 5, 6, 10, 11, 12	48	64	30	8	0.75	0.72
7	3, 5, 6, 8, 10, 11, 12	49	58	36	7	0.71	0.7
8	3, 5, 6, 8, 9, 10, 11, 12	48	65	29	8	0.75	0.72
9	3, 5, 6, 7, 8, 9, 10, 11, 12	49	63	31	7	0.75	0.72
10	3, 5	27	89	5	29	0.77	0.61
11	3, 5, 10	46	60	34	10	0.71	0.68
12	2, 3, 5, 10	32	82	24	12	0.76	0.64
13	2, 3, 5, 10, 12	45	64	30	11	0.73	0.69
14	9, 10	23	82	12	33	0.7	0.51
15	5, 9, 10	45	58	36	11	0.69	0.66
16	3, 5, 6, 9, 10	43	67	27	13	0.73	0.68
17	3, 5, 6, 9, 10, 12	43	68	26	13	0.74	0.69

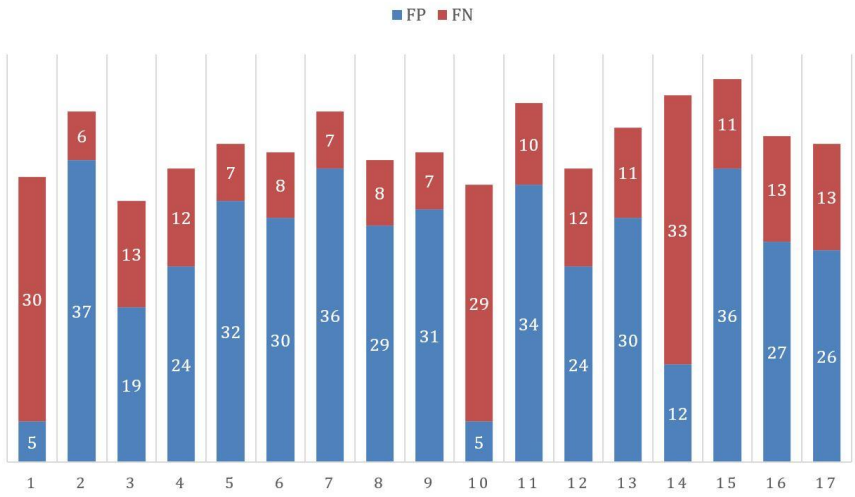


Рис. 1. Диаграмма ложных ответов различных ансамблей
Fig. 1. Diagram of false responses of different ensembles

Нетрудно увидеть (см. табл. 1), что полученный ансамбль имеет меньшее число ошибок, чем все одиночные алгоритмы, кроме случайного леса деревьев решений. Но последний, в свою очередь, является «небезопасным» алгоритмом с очень большим значением FN. Таким образом, полученный ансамбль действительно является наиболее хорошим решением из полученных в данной работе.

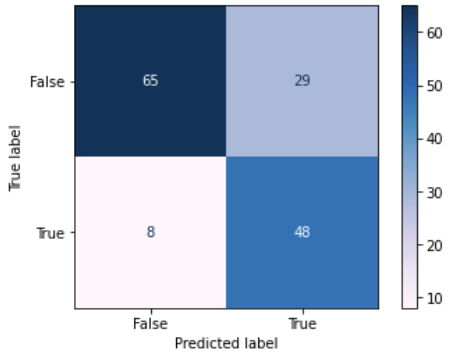
В итоговый ансамбль вошли следующие модели:

1) Машина опорных векторов с полиномиальным ядром третьего порядка;
2) Машина опорных векторов с линейным ядром;
3) Машина опорных векторов с радиально-базисной функцией в качестве ядра;
4) Логистическая регрессия;
5) Классификатор на линейном дискриминантном анализе;
6) Случайный лес деревьев решений на 64 деревьях;
7) Наивный байесовский классификатор;
8) Многослойный перцептрон с двумя скрытыми слоями размерами 64 и 16 нейронов с функцией активации ReLU.

В первыых четырех моделях используется балансирование весов классов, так как исходная выборка не сбалансирована (записей из класса «нет лавины» в несколько раз больше записей класса «есть лавина»).

Верификация модели

Проверка работы полученного алгоритма многократно проводилась на валидационной выборке на каждой итерации настройки алгоритма (рис. 2) и один контрольный раз на тестовой выборке. Также были вычислены средние значения метрик (macro avg – среднее арифметическое значение, weighted avg – взвешенное среднее арифметическое пропорционально размерам классов). Разбиение на такие выборки необходимо, чтобы исключить попадание в модель метаинформации, которая неизбежно используется при подгонке параметров модели на валидационной выборке. Поэтому результат, полученный на тестовой выборке (рис. 3), можно считать итоговым, так как модель ни напрямую, ни косвенно не обучалась на данных из тестовой выборки.

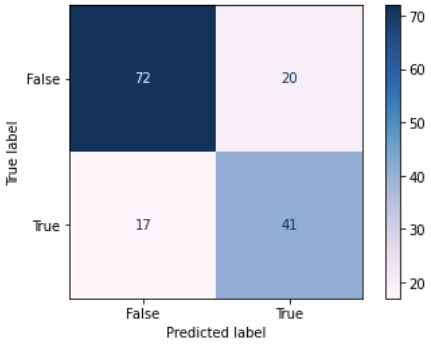


а

	precision	recall	f1-score
False	0.89	0.69	0.78
True	0.62	0.86	0.72
accuracy			0.75
macro avg	0.76	0.77	0.75
weighted avg	0.79	0.75	0.76

б

Рис. 2. Матрица предсказаний алгоритма (*а*) и метрики (*б*) на валидационной выборке
Fig. 2. Confusion matrix (*a*) and metrics (*b*) on validation dataset



а

	precision	recall	f1-score
False	0.81	0.78	0.80
True	0.67	0.71	0.69
accuracy			0.75
macro avg	0.74	0.74	0.74
weighted avg	0.76	0.75	0.75

б

Рис. 3. Матрица предсказаний алгоритма (*а*) и метрики (*б*) на тестовой выборке
Fig. 3. Confusion matrix (*a*) and metrics (*b*) on test dataset

Заключение

В ходе проделанной работы были собраны, сгенерированы, очищены и приведены к рабочему виду данные. Обучено множество моделей машинного обучения и составлен ансамбль из алгоритмов, имеющий хорошие показатели в предсказании схода лавины по прогнозу погоды. Стоит отметить, что полученное решение использует лишь данные о метеорологической обстановке в горном районе и не использует реальные данные о состоянии снежного покрова и экспертные данные из лавинных центров. Это позволяет использовать полученное решение в горных районах, не имеющих развитых лавинных центров, при условии наличия статистики схода лавин в районе за срок в несколько лет и показаний метеорологических станций. Стоит отметить эффективность генерирования состояния снежного покрова с помощью программы SNOWPACK, которая подтверждается улучшением качества работы алгоритма предсказания лавины при добавлении сгенерированных данных.

В итоге было получено решение, имеющее предиктивную силу. Это свидетельствует о том, что предсказание лавин по метеорологическим данным и сгенерированным данным о состоянии снежного покрова возможно.

Список литературы

1. **Bahram Choubin**, **Moslem Borji**, **Amir Mosavi**, **Farzaneh Sajedi-Hosseini**, **Vijay P. Singh**, **Shahaboddin Shamshirband**. Snow avalanche hazard prediction using machine learning methods. *Journal of Hydrology*, 2019, vol. 577, p. 123929. DOI 10.1016/j.jhydrol.2019.123929
2. **Pozdnoukhov A.**, **Purves R. S.**, **Kanevski M.** Applying machine learning methods to avalanche forecasting. *Annals of Glaciology*, 2008, vol. 49. DOI 10.3189/172756408787814870
3. **Марченко Е. С.** Пространственная оценка устойчивости снежного покрова для определения возможности схода лавин разных генетических типов: Автореф. дис. … канд. геогр. наук / Моск. гос. ун-т. М., 2013.
4. **Schirmer M.**, **Lehning M.**, **Schweizer J.** Statistical forecasting of regional avalanche danger using simulated snow-cover data. *Journal of Glaciology*, 2009, vol. 55, no. 193. DOI 10.3189/002214309790152429
5. **Cordy P.**, **McClung D. M.**, **Hawkins C. J.**, **Tweedy J.**, **Weick T.** Computer assisted avalanche prediction using electronic weather sensor data. DOI 10.1016/j.coldregions.2009.07.006

6. **Радеев Н. А.** Сбор и предобработка данных для системы предсказания лавинной опасности // Тез. докл. Междунар. конф. «Мальцевские чтения 2020». Новосибирск, 2020. С. 91.
7. **Головин Н. А.**, **Савин Н. П.**, **Яхъяева Г. Э.** Применение методов машинного обучения для структурирования базы прецедентов компьютерных атак // Материалы Междунар. конф. «Знания – Онтологии – Теории». Новосибирск, 2019. С. 122–128.

References

1. **Bahram Choubin**, **Moslem Borji**, **Amir Mosavi**, **Farzaneh Sajedi-Hosseini**, **Vijay P. Singh**, **Shahaboddin Shamshirband**. Snow avalanche hazard prediction using machine learning methods. *Journal of Hydrology*, 2019, vol. 577, p. 123929. DOI 10.1016/j.jhydrol.2019.123929
2. **Pozdnoukhov A.**, **Purves R. S.**, **Kanevski M.** Applying machine learning methods to avalanche forecasting. *Annals of Glaciology*, 2008, vol. 49. DOI 10.3189/172756408787814870
3. **Marchenko E. S.** Spatial assessment of the stability of snow cover to determine the possibility of avalanches of different genetic types. Author. Dis. ... Cand. Geogr. Sciences. Moscow State University. Moscow, 2013. (in Russ.)
4. **Schirmer M.**, **Lehning M.**, **Schweizer J.** Statistical forecasting of regional avalanche danger using simulated snow-cover data. *Journal of Glaciology*, 2009, vol. 55, no. 193. DOI 10.3189/002214309790152429
5. **Cordy P.**, **McClung D. M.**, **Hawkins C. J.**, **Tweedy J.**, **Weick T.** Computer assisted avalanche prediction using electronic weather sensor data. DOI 10.1016/j.coldregions.2009.07.006
6. **Radeev N. A.** Collection and preprocessing of data for the avalanche hazard prediction system. In: Abstracts of the International Conference “Maltsev Readings 2020”. Novosibirsk, 2020, p. 91. (in Russ.)
7. **Golovin N. A.**, **Savin N. P.**, **Yakhyaeva G. E.** Application of machine learning methods to structure the base of computer attacks precedents. In: Materials of the International Conference “Knowledge – Ontology – Theory”. Novosibirsk, 2019, p. 122–128. (in Russ.)

Материал поступил в редколлегию
Received
11.03.2021

Сведения об авторе

Радеев Никита Андреевич, бакалавр кафедры общей информатики факультета информационных технологий Новосибирского государственного университета (Новосибирск, Россия)

n.radeev@g.nsu.ru

Information about the Author

Nikita A. Radeev, Bachelor of General Informatics Department, Faculty of Information Technologies, Novosibirsk state University (Novosibirsk, Russian Federation)

n.radeev@g.nsu.ru

УДК 004.415
DOI 10.25205/1818-7900-2021-19-2-102-114

Автоматизированная система синтеза структурированного учебного контента

Е. А. Сидорова, А. В. Долгова, С. П. Железняк

Омский государственный университет путей сообщения
Омск, Россия

Аннотация

Изучение дисциплины «Информатика» на современном этапе практически невозможно представить без применения электронных образовательных ресурсов. В качестве этих ресурсов чаще всего выступают электронные учебно-методические комплексы (ЭУМК) дисциплины. Формирование комплекта учебных материалов для наполнения ЭУМК – трудоемкий и ресурсоемкий процесс, от результатов которого зависит эффективность работы студентов на занятии. Статья посвящена разработке автоматизированной системы синтеза структурированного учебного контента, представляющей собой универсальную оболочку ЭУМК дисциплины «Информатика». Система позволяет унифицировать подход к учебному процессу, применять ее вне зависимости от трудоемкости и содержания дисциплины, обеспечивать эффективную работу студентов в локальной вычислительной сети (ЛВС) вуза. Разработанная система включает в себя шесть независимых модулей, реализующих выбор параметров для загрузки комплекса, сервисные функции, загрузку ЭУМК, авторизацию и настройки, контроль формирования компетенций. В статье раскрыта концепция работы каждого модуля. Подробно описана работа модуля выбора параметров загрузки ЭУМК, приведена укрупненная графическая схема алгоритма его функционирования. Структура ЭУМК реализована в соответствии с рабочей программой дисциплины «Информатика». Весь учебный контент разделен на два логически завершенных раздела, каждый из которых включает в себя несколько подразделов. В качестве разделов и подразделов ЭУМК выступают узлы иерархического дерева TreeView, которые наполняются по специальному алгоритму элементами учебного контента. При наполнении ЭУМК в подраздел контрольно-измерительных материалов подключаются специально разработанные на языке VBA электронные интерактивные тренажеры – программы-генераторы заданий по разным темам, а также универсальная тестово-обучающая программа контроля знаний. Рассмотренная в статье система обладает следующими достоинствами: небольшой объем занимаемой на диске памяти, возможность работы как в ЛВС с выделенным сервером, так и с хранилищем на локальном компьютере студента, гибкое наполнение заявленных разделов в зависимости от объема рассматриваемого курса, уровня подготовки студентов.

Ключевые слова

автоматизированная система, электронный учебно-методический комплекс, модуль, интерфейс, графическая схема алгоритма

Для цитирования

Сидорова Е. А., Долгова А. В., Железняк С. П. Автоматизированная система синтеза структурированного учебного контента // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 102–114. DOI 10.25205/1818-7900-2021-19-2-102-114

Automated System Synthesis of Structured Educational Content

E. A. Sidorova, **A. V. Dolgova**, **S. P. Zheleznyak**

Omsk State Transport University
Omsk, Russian Federation

Abstract

At present the study of computer science is almost impossible imagine without electronic educational resources usage. These resources are most often represents at learning package. Synthesis of learning packages is labour and resource intensive process. The results of this process influence to students work efficient. Article is devoted to creating automated system synthesis of structured educational content. This system is the universal shell for computer science learning package. This system allow to unify educational process approach, apply regardless of course content, increase student work efficient in university local network. Developed system is include six independent module. These modules realize system load properties choose, service, system load, authentification and tools, student competence control. This article contains modules work concept. Module work concept of system load properties choose is describes in detail and it's function algorithm is presented. Educational content is structured to computer science teaching program. All content is divided to two logical section. Which section is contains several subsection. Those section and subsection are TreeView hierarchical tree nodes. They are included educational content elements by special algorithm. Universal test system and special developed on VBA interactive trainers are included to competence control subsection. Those trainers are program's which can generate task for several topic. The system which considering in this article have some advantages. It's occupied small memory on hard disk, it's worked in network and on local computer. The educational content can be flexible included in this system without on course volume and student basic knowledge level.

Keywords

automated system, learning package, module, interface, algorithm

For citation

Sidorova E. A., Dolgova A. V., Zheleznyak S. P. Automated System Synthesis of Structured Educational Content. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 102–114. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-102-114

Введение

Учебный процесс в высшем учебном заведении на современном этапе невозможно представить без применения информационных технологий (ИТ) [1]. Федеральные государственные образовательные стандарты высшего образования (ФГОС ВО) последнего поколения предоставляют право образовательной организации применять электронное обучение при реализации программ бакалавриата и специалитета.

Электронное обучение предусматривает непосредственное применение ИТ для создания и передачи электронных образовательных ресурсов (ЭОР) субъектам учебного процесса. Одним из основополагающих аспектов при реализации электронного обучения является создание электронной информационно-образовательной среды [1], обеспечивающей доступ к ЭОР, которые чаще всего представляются в виде электронных учебно-методических комплексов (ЭУМК).

Кафедра «Информатика и компьютерная графика» (далее кафедра) Омского государственного университета путей сообщения (ОмГУПС) осуществляет образовательную деятельность более чем по 70 учебным планам по восьми специальностям и 18 направлениям подготовки. Все они предусматривают проведение лекций, лабораторных работ, контроля самостоятельной работы (КСР), промежуточной аттестации по дисциплине «Информатика». При этом трудоемкость освоения дисциплины (количество академических часов) и перечень целевых компетенций в учебных планах разных специальностей и направлений подготовки

[1] Российская Федерация. Законы. Об образовании в Российской Федерации : Федеральный закон № 273-ФЗ : [принят Государственной думой 29 декабря 2012 года : одобрен Советом Федерации 26 декабря 2012 года]. Доступ из СПС «КонсультантПлюс».

различаются, в связи с чем содержание изучаемого курса может существенно варьироваться. Поскольку ФГОС ВО не предъявляет конкретных требований к содержанию дисциплины, оно может также корректироваться в зависимости от уровня базовой подготовки студентов, индивидуальных особенностей организации курса преподавателем, профессиональной востребованности будущих выпускников.

Традиционно лекции для потока студентов (далее поток) читает ведущий преподаватель (далее лектор), а лабораторные работы и занятия КСР могут проводить как сам лектор, так и другие педагоги, в том числе лекторы других потоков (далее преподаватель). Как правило, преподаватель осуществляет учет успеваемости в закрепленных группах и по результатам занятий предоставляет лектору сведения о полноте и качестве учебно-методических материалов (УММ) на основе анализа ошибок, совершаемых студентами во время выполнения заданий.

При изучении информатики невозможно осуществлять учебный процесс, применяя только традиционные учебно-методические пособия. Для результативной работы студентов на занятиях необходимы шаблоны документов, заготовки исходных данных для выполнения заданий, видеоматериалы и т. п. Поскольку комплект материалов становится довольно обширным, без его четкого структурирования эффективная работа невозможна, особенно с учетом того, что информатика обычно изучается на первом курсе, когда студенты еще не имеют достаточно развитых навыков организации своей учебной деятельности. С целью решения данной проблемы на сервере кафедры в локальной вычислительной сети (ЛВС) университета организована следующая структура взаимодействия субъектов учебного процесса (рис. 1). В каталоге лектора располагаются задания, предназначенные для выполнения студентами, общие требования по их оформлению и защите, учебно-методические разработки, применяемые в учебном процессе. Для преподавателя организован персональный каталог, в котором, как правило, размещены ярлык на каталог лектора, файлы учета успеваемости закрепленных за преподавателем групп, отдельные папки для отчетов студентов каждой учебной группы.

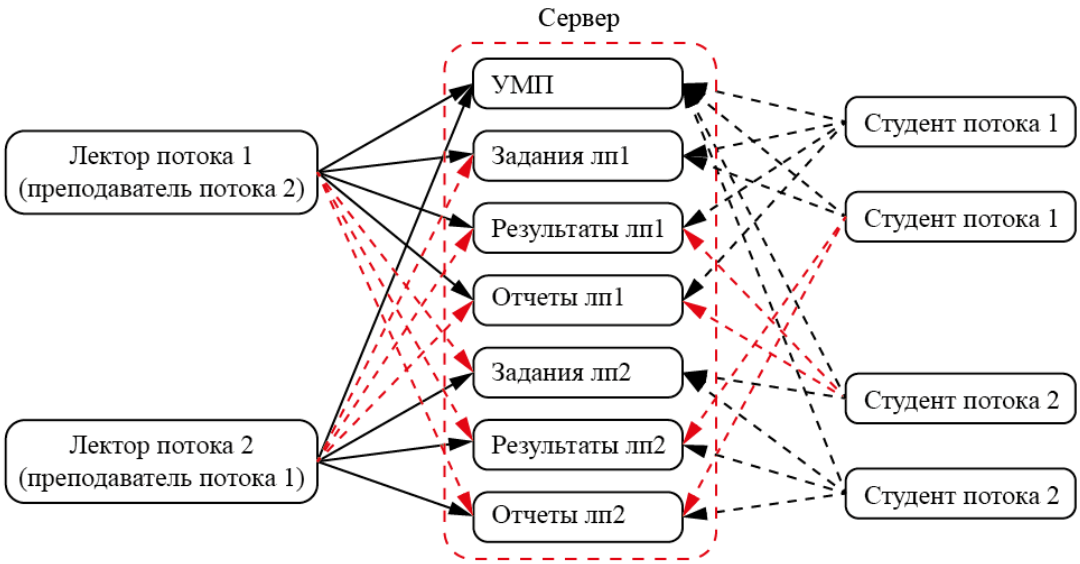


Рис. 1. Структура сетевого взаимодействия субъектов учебного процесса
Fig. 1. The educational process subject network interaction structure

Глубина вложенности структуры каталогов и файлов может быть достаточно большой, что создает для студентов трудности в оперативной навигации, особенно когда не все виды занятий проводит лектор. Дублирование учебных материалов лектора в каталоги преподавателей этой проблемы не решает, так как приводит к нерациональному использованию памяти сервера и необходимости постоянного контроля своевременного обновления материалов при их корректировке. Таким образом, актуальной становится задача разработки автоматизированной системы [2], представляющей собой универсальную оболочку ЭУМК, которая позволяет синтезировать персонализированный комплект электронных учебно-методических материалов для любого субъекта учебного процесса независимо от трудоемкости и содержания дисциплины.

Архитектура автоматизированной системы

На основе накопленного педагогического опыта, анализа научной и учебно-методической литературы, изучения современных образовательных технологий представления учебного материала на кафедре разработана и внедрена в учебный процесс автоматизированная система синтеза структурированного учебного контента дисциплины «Информатика» (рис. 2), позволяющая поэтапно сформировать индикаторы достижения компетенций, предусмотренные ФГОС ВО.

Рис. 2. Интерфейс автоматизированной системы
Fig. 2. Automated system interface to systhesis of structured educational content

Согласно статистическим данным за период с февраля 2020 г. по февраль 2021 г., около 75 % всех стационарных компьютеров используют операционную систему Windows [2], ввиду чего она выбрана платформой для разработки системы. Программная реализация выполнена на языке Visual Basic, который по состоянию на март 2021 г. занимает шестую позицию в рейтинге популярности языков программирования TIOBE [3]. Структура системы, синтезирующей ЭУМК, выполнена таким образом, что позволяет работать как в компьютерной сети кафедры, так и на локальном компьютере студента.

Архитектура разработанной системы содержит шесть модулей (рис. 3).

[2] Desktop Operating System Market Share Worldwide // StatCounter. URL: https://gs.statcounter.com/os-market-share/desktop/worldwide (дата обращения 20.04.2021).

[3] TIOBE Programming Community index // TIOBE [сайт]. URL: https://www.tiobe.com/tiobe-index/ (дата обращения 20.04.2021).

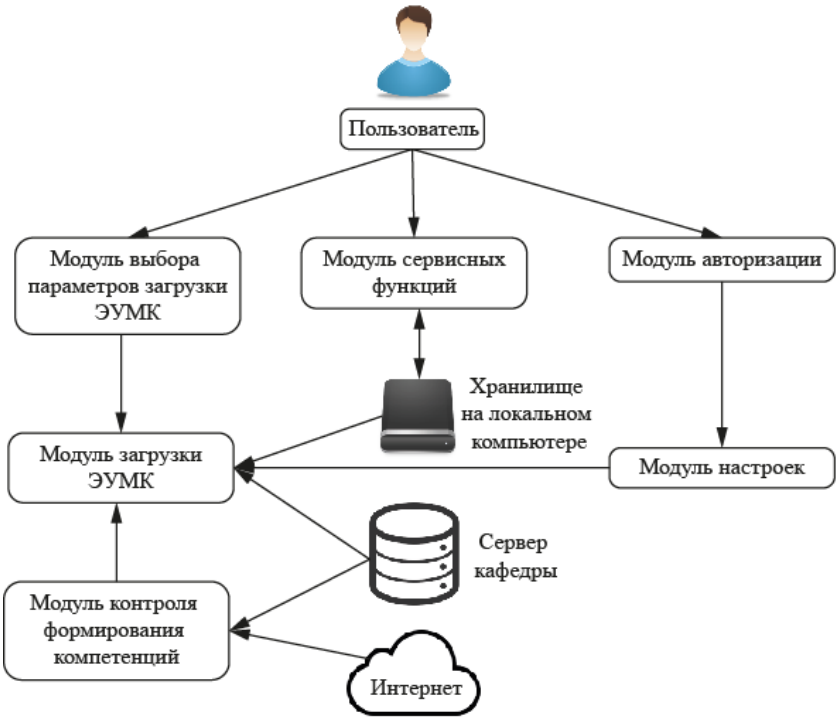


Рис. 3. Архитектура автоматизированной системы
Fig. 3. Automated system architecture

1. Модуль выбора параметров загрузки ЭУМК позволяет пользователю задать необходимые опции для работы с материалами.
2. Модуль загрузки ЭУМК отвечает за наполнение его учебными материалами с сервера или из локального хранилища.
3. Модуль контроля отвечает за доступ к системе тестирования и разработанным компьютерным тренажерам.
4. Модуль сервисных функций выполняет резервное копирование и создает zip-архив с материалами ЭУМК, либо считывает zip-архив из локального хранилища и распаковывает его в директорию исполняемого файла.
5. Модуль авторизации необходим для доступа к настройкам параметров загрузки ЭУМК.
6. Модуль настроек содержит в себе сведения о конфигурации системы.

Модуль выбора параметров загрузки ЭУМК

Интерфейс автоматизированной системы содержит горизонтальное выпадающее меню с тремя пунктами: «Материалы и задания», «Сервис», «Выход».

Для обеспечения унифицированного подхода к учебному процессу и облегчения студентам навигации в ЛВС кафедры служит модуль выбора параметров загрузки ЭУМК. Он вызывается пользователем в пункте меню «Материалы и задания» и представляет собой форму, на которой размещаются три контейнера с переключателями (рис. 4). В этих контейнерах согласно алгоритму, представленному на рис. 5, размещается информация из текстового файла настроек, в котором перечислены изучающие дисциплину потоки, а также фамилии лекторов и преподавателей, работающих с этими потоками. При обработке файла настроек

все указанные выше данные считываются в текстовый массив lines, который разделяется на три массива. Каждый элемент выделенного массива проверяется на уникальность, что позволяет избежать дублирований переключателей в отдельном контейнере.

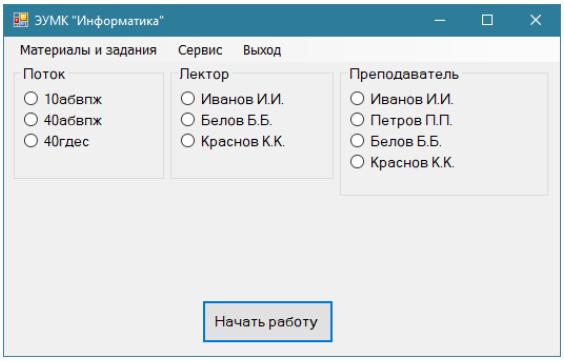


Рис. 4. Интерфейс автоматизированной системы
в режиме выбора параметров загрузки ЭУМК
Fig. 4. Automated system interface of load properties choose mode

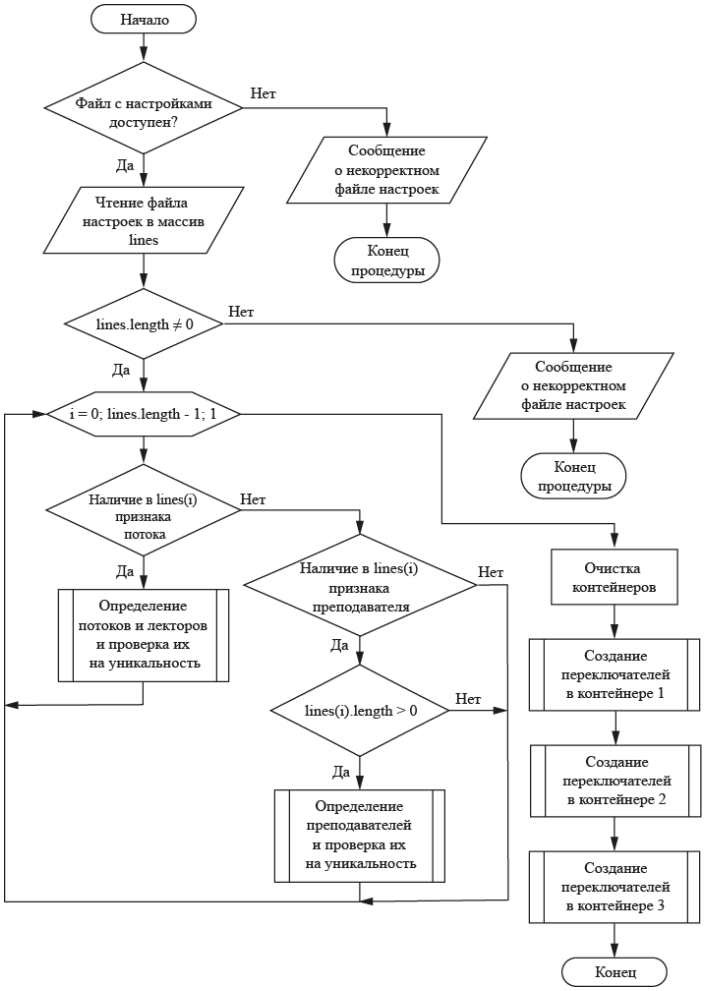


Рис. 5. Укрупненная графическая схема
алгоритма работы модуля выбора параметров загрузки ЭУМК
Fig. 5. Function algorithm of load properties choose module

Модуль загрузки ЭУМК

После выбора пользователем соответствующих параметров в окне, представленном на рис. 4, включается индикатор процесса заполнения ЭУМК документами и начинает функционировать модуль загрузки. С целью обеспечения студентам доступа к материалам во время самостоятельной работы в домашних условиях организована возможность функционирования модуля загрузки ЭУМК в двух режимах. При запуске исполняемого файла системы с сервера кафедры производится синтез персонализированного комплекта учебных материалов из соответствующих каталогов лектора и преподавателя, а также подключаются каталоги с УММ. Если запуск исполняемого файла производится с локального компьютера студента, то ЭУМК наполняется данными из служебных каталогов, находящихся в директории с этим файлом.

Поясним функционирование автоматизированной системы на примере изучения двухсеместрового курса информатики по специальности 23.05.03 «Подвижной состав железных дорог». В соответствии с рабочей программой дисциплины разработанный ЭУМК разделен на два логически завершенных раздела. Для эффективного осуществления навигации по ЭУМК и представления информации в нем в виде иерархического дерева выбран элемент TreeView. Это дерево содержит два корневых узла, каждый из которых включает в себя шесть дочерних узлов (подразделов). В качестве подразделов выступают УММ, задания для выполнения курса лабораторных работ, контрольно-измерительные материалы для проверки уровня сформированности индикаторов достижения компетенций (рис. 6). Эти подразделы наполняются заголовками каталогов и документов, размещенных в соответствующих директориях (см. рис. 6). Считывание вложенных каталогов в дочерний узел реализовано рекурсивной процедурой [3].

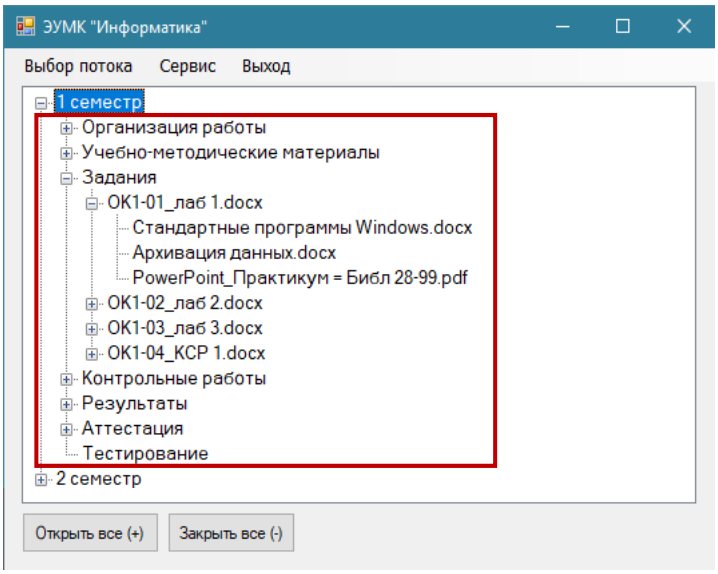


Рис. 6. Интерфейс раздела «Материалы и задания»
Fig. 6. Section “Materials and tasks” interface

Для унификации требований к выполнению заданий и их контролю в рамках одного потока студентов подразделы «Организация работы», «Задания», «Контрольные работы» и «Аттестация» наполняются из каталога лектора. Для отображения успеваемости студентов одного потока, занимающихся в разных подгруппах, подраздел «Результаты» заполняется из

директории соответствующего преподавателя. Таким образом, студентам, обучающимся у разных преподавателей, обеспечивается персонализированный доступ к учебным материалам и результатам работы.

Для каждого занятия обучающимся необходим уникальный комплект УММ. Эффективно представить его позволяет многоярусная навигация. Для ее организации при заполнении подраздела «Задания» к каждой работе динамически подключаются необходимые внешние файлы, шаблоны и УММ по ссылке из Word-документа с заданием. Имена внешних файлов и шаблонов формируются по специальной маске, по которой осуществляется контекстный поиск. Для добавления в каждую работу учебно-методических материалов в качестве ссылки применяется специальное структурированное имя. Найденные документы подключаются в качестве дочерних узлов к текущей работе (см. рис. 6).

Модуль контроля

Подразделы «Контрольные работы» и «Тестирование» заполняются при работе модуля контроля. В подраздел «Контрольные работы» включаются разработанные на кафедре электронные интерактивные тренажеры по темам «Измерение информации», «Алгебра логики» и «Запись арифметических выражений на VBA», задания для выполнения контрольных работ по темам «Программные средства реализации информационных процессов», «Основы программирования».

Электронные интерактивные тренажеры по темам «Измерение информации» (далее ЭИТИИ), «Алгебра логики» (далее ЭИТАЛ) и «Запись арифметических выражений на VBA» (далее ЭИТАВ) реализованы в среде Microsoft Excel на языке программирования Visual Basic for Applications (рис. 7–9). Тренажеры представляют собой программы-генераторы заданий [4] по указанным темам и имеют интуитивно понятный интерфейс.

ТРЕНАЖЕР по теме "ИЗМЕРЕНИЕ ИНФОРМАЦИИ"

Начать | Завершить | Очистить

		Пуск 1	Начало	Конец	Мин.	Итог
			9:55:56	10:12:33	16,6	28
	Результаты вводить в ЗЕЛЕНЫЕ ячейки!	Результат	Ответ	+/-	Тип задачи	Балл
1	Сколько байт информации содержит сообщение, составленное из 192 символов 256-символьного алфавита?	1536	192	-	Алф - 16 б.	0
2	На предприятии работают 10 инженеров и 30 рабочих. Какое количество информации содержит сообщение о том, что в отпуск ушел один инженер?	2	2	+	Вер - 18 б.	18
3	Аэропорт за сутки принимает несколько рейсов самолетов. Сообщение о том, что прибыл рейс № 256 содержит 9 бит информации. Сколько всего рейсов самолетов принимает аэропорт за сутки?	512	512	+	Сод - 10 б.	10

Рис. 7. Интерфейс ЭИТИИ
Fig. 7. Electronic interactive simulator “Information measurement” interface

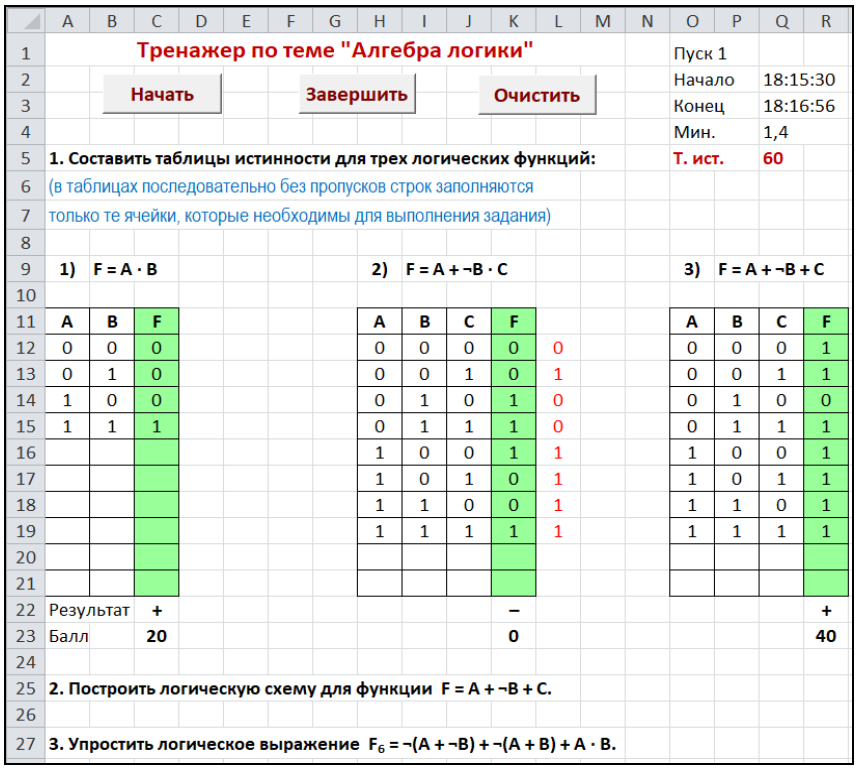


Рис. 8. Интерфейс ЭИТАЛ
Fig. 8. Electronic interactive simulator “Boolean algebra” interface

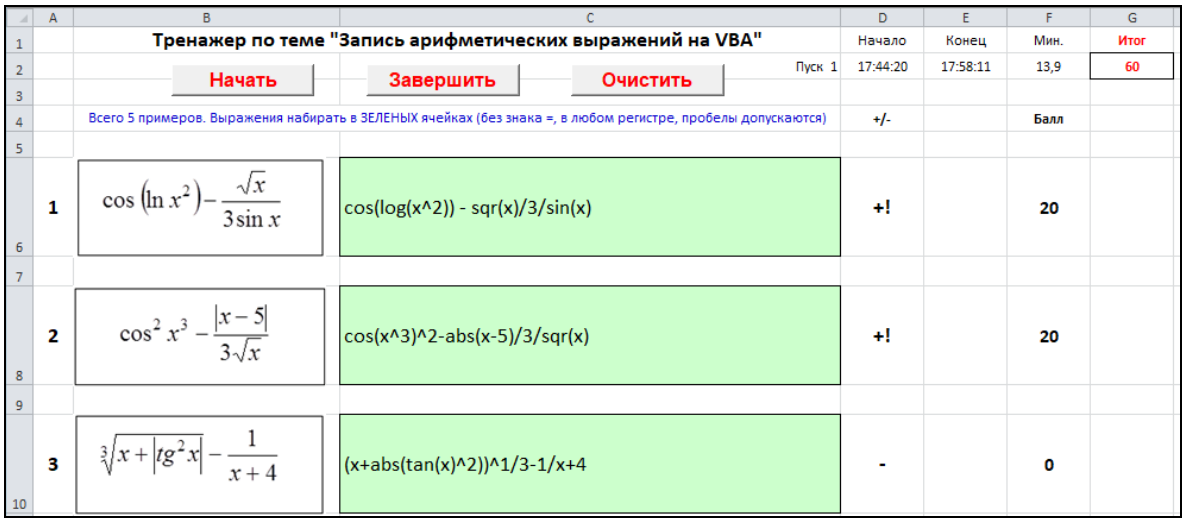


Рис. 9. Интерфейс ЭИТАВ
Fig. 9. Electronic interactive simulator “Arithmetic expressions on VBA” interface

Набор заданий в каждом сеансе работы указанных тренажеров формируется случайным образом по специальному алгоритму, обеспечивающему заданную тематическую структуру и пропорциональное наличие вопросов разного типа и сложности, и включает все типы задач по указанным темам. Для ввода пользователем результатов вычислений в бланке заданий

предусмотрены специальные ячейки, корректировка сформированного набора задач запрещена. После завершения сеанса работы результаты в ячейках проверяются автоматически. Статистика выполнения каждого задания отображается на листе в виде специального протокола. Все результаты пользователя сохраняются в отдельном документе и могут использоваться для мониторинга, накопления статистики и дальнейшего анализа (индивидуально по каждому студенту или по группе в целом), например, определения качества тренинговых заданий [5; 6] или построения графических зависимостей, отображающих результативность решения заданий определенного типа. Доступ ко всем тренажерам возможен как при работе в компьютерной сети кафедры, так и на локальном компьютере студента.

При работе модуля контроля к подразделу «Тестирование» подключается разработанная на кафедре официально зарегистрированная в Реестре программ для ЭВМ «Универсальная тестово-обучающая программа контроля профессиональных знаний Test Master 2007 (версия 1.0)» [4]. Активация данного подраздела при работе в ЛВС кафедры запускает исполняемый файл программы тестирования. Для подготовки студента к компьютерному тестированию с его локального компьютера при наличии Интернета [7] активация рассматриваемого подраздела запускает специальный алгоритм, предоставляющий доступ к порталу дистанционного обучения ОмГУПСа.

Модуль сервисных функций

В пункте меню «Сервис» размещены инструкция по работе с автоматизированной системой, базовые настройки, опции резервного копирования и восстановления материалов, сведения о программе.

Для резервного копирования материалов ЭУМК и их восстановления на локальном компьютере студента предусмотрен специальный программный модуль сервисных функций, который активируется при выборе в пункте меню «Сервис» опций «Сохранить» или «Восстановить» (рис. 10). В связи с тем, что полный комплект УММ занимает значительный объем памяти, при резервном копировании ЭУМК студенту предоставляется возможность выбора типа копирования: с УММ или без них. Для ускорения резервного копирования студент может выбрать конкретный семестр для сохранения. Для восстановления на локальном компьютере актуальной версии ЭУМК имя созданного zip-архива содержит номер выбранного семестра, дату и время сохранения данных.

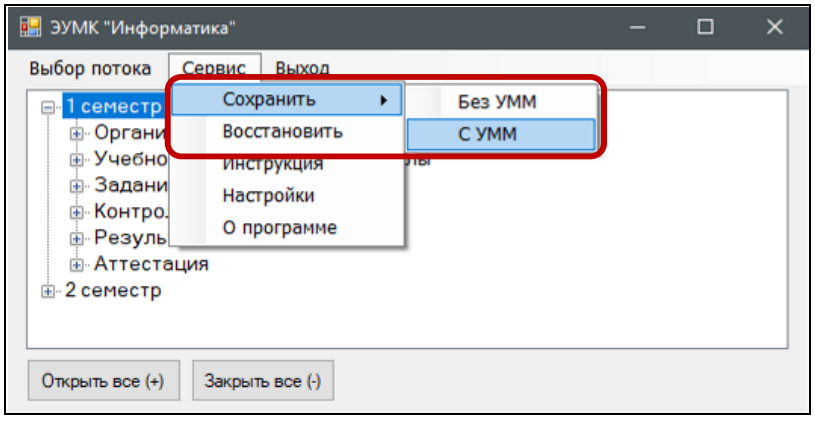


Рис. 10. Меню сервисных функций
Fig. 10. Service functions menu

[4] Свидетельство о государственной регистрации программы для ЭВМ № 2008614627 Российская Федерация. Универсальная тестово-обучающая программа контроля профессиональных знаний Test Master 2007 : № 200861340 : заявл. 29.07.2008 : зарегистр. 25.09.2008 / И. Л. Саля, Е. А. Сидорова ; заявитель ОмГУПС. 1 с. : ил.

Результаты и выводы

Представленная автоматизированная система обеспечивает синтез структурированного учебного контента дисциплины «Информатика» в строгом соответствии с потребностями обучающихся и за время ее использования зарекомендовала себя положительно. Основными достоинствами системы являются небольшой объем занимаемой на диске памяти, возможность работы как в ЛВС с выделенным сервером, так и с хранилищем на локальном компьютере студента, удобный интерфейс, гибкое наполнение заявленных разделов в зависимости от объема рассматриваемого курса и уровня подготовки студентов [8].

В перспективе планируется развитие системы по следующим направлениям: подключение модуля сбора статистических данных, поддержка облачных сервисов (таких как Google Диск, Яндекс.Диск, Dropbox) при работе модуля загрузки ЭУМК, автоматическая генерация заданий различного уровня сложности и программная проверка их выполнения.

Список литературы

1. **Safiullin N.**, **Maratkanova E.**, **Safiullin L.**, **Saipullaev U.** The Global Information Educational Resources: Methodological Issues. *Procedia – Social and Behavioral Sciences*, 2015, vol. 191, p. 2391–2395.
2. **González A. G.**, **Salgado D. R.**, **García Sanz-Calcedo J.**, **Cruz García C.**, **Barrios Muriel J.**, **Lopez Perez O.**, **Alvarez Garsia F. J.** A teaching methodology for the real-time assessment of students' competencies related to manufacturing subjects using technology based on electronic devices. *Procedia Manufacturing*, 2019, vol. 41, p. 579–586.
3. **Шарипов М. И.** Организация вызовов рекурсивных функций в современном программном обеспечении и проблемы производительности рекурсивных алгоритмов // Актуальные проблемы гуманитарных и естественных наук. 2016. № 5-1. С. 122–126.
4. **Посов И. А.** Программирование генераторов задач // Компьютерные инструменты в образовании. 2010. № 3. С. 19–31.
5. **Будилова А. С.** Развитие коммуникативных умений студентов при обучении информационным технологиям на основе тренинга // Наука и перспективы. 2016. № 4. URL: https://cyberleninka.ru/article/n/razvitie-kommunikativnyh-umeniy-studentov-pri-obuchenii-informatsionnym-tehnologiyam-na-osnove-treninga/viewer (дата обращения 20.04.2021).
6. **Киселева Н. А.** К вопросу об использовании тренинговых занятий в образовательном процессе вуза // Педагогические науки. URL: https://cyberleninka.ru/article/n/k-voprosu-ob-ispolzovanii-treningovyh-zanyatiy-v-obrazovatelnom-protsesse-vuza/viewer (дата обращения 20.04.2021).
7. **Евсеева Т. П., Сабирова Ю. В.** Разработка тестовых заданий как один из методов технологии интерактивного обучения // Вестник Казан. технол. ун-та. 2014. Т. 17, № 5. С. 320–324.
8. **De la Peña Esteban F. D.**, **Lara Torralbo J. A.**, **Lizcano Casas D.**, **Martínez Rey M. A.** Expert system for problem solving in distance university education: The successful case of the subject "operations management". *Journal Expert Systems*, 2019, vol. 36 (5), e12444.

References

1. **Safiullin N.**, **Maratkanova E.**, **Safiullin L.**, **Saipullaev U.** The Global Information Educational Resources: Methodological Issues. *Procedia – Social and Behavioral Sciences*, 2015, vol. 191, p. 2391–2395.
2. **González A. G.**, **Salgado D. R.**, **García Sanz-Calcedo J.**, **Cruz García C.**, **Barrios Muriel J.**, **Lopez Perez O.**, **Alvarez Garsia F. J.** A teaching methodology for the real-time assess-

ment of students' competencies related to manufacturing subjects using technology based on electronic devices. *Procedia Manufacturing*, 2019, vol. 41, p. 579–586.
3. **Sharipov M. I.** Organization of calls of recursive functions in modern software and problems of performance of recursive algorithms. *Actual problems of the humanities and natural sciences*, 2016, no. 5-1, p. 122–126. (in Russ.)
4. **Posov I. A.** Programming of problem generators. *Computer tools in education*, 2010, no. 3, p. 19–31. (in Russ.)
5. **Budilova A. S.** Development of students' communication skills in teaching information technologies based on training. *Science and Perspectives*, 2016, no. 4. (in Russ.) URL: https://cyberleninka.ru/article/n/razvitie-kommunikativnyh-umeniy-studentov-pri-obuchenii-informatsionnym-tehnologiyam-na-osnove-treninga/viewer (accessed 20.04.2021).
6. **Kiseleva N. A.**To the question of the use of training sessions in the educational process of the university. *Pedagogical sciences*. (in Russ.) URL: https://cyberleninka.ru/article/n/k-voprosu-ob-ispolzovanii-treningovyh-zanyatiy-v-obrazovatelnom-protsesse-vuza/viewer (accessed 20.04.2021).
7. **Evseeva T. P.** Development of test tasks as one of the methods of interactive learning technology / T. P. Evseeva, Yu. V. Sabirova. *Bulletin of the Kazan Technological University*, 2014, vol. 17, no. 5, p. 320–324.
8. **De la Peña Esteban F. D.**, **Lara Torralbo J. A.**, **Lizcano Casas D.**, **Martínez Rey M. A.** Expert system for problem solving in distance university education: The successful case of the subject "operations management". *Journal Expert Systems*, 2019, vol. 36 (5), e12444.

Материал поступил в редколлегию
Received
09.05.2021

Сведения об авторах

Сидорова Елена Анатольевна, доктор технических наук, доцент, заведующий кафедрой «Информатика и компьютерная графика», ФГБОУ ВО «Омский государственный университет путей сообщения» (Омск, Россия)

armsid@mail.ru
ResearcherID B-5089-2017
ORCID 0000-0001-5312-7564

Долгова Анна Владимировна, кандидат технических наук, доцент, доцент кафедры «Информатика и компьютерная графика», ФГБОУ ВО «Омский государственный университет путей сообщения» (Омск, Россия)

9059238462@mail.ru
ResearcherID D-9708-2019
ORCID 0000-0002-8201-8823

Железняк Светлана Петровна, кандидат технических наук, доцент кафедры «Информатика и компьютерная графика», ФГБОУ ВО «Омский государственный университет путей сообщения» (Омск, Россия)

zhsp120866@yandex.ru
ResearcherID AAM-5444-2021
ORCID 0000-0001-8957-2678

Information about the Authors

Elena A. Sidorova, Doctor of Technical Sciences, Assistant professor, Head of department "Computer Science and Computer graphics", Omsk State Transport University (Omsk, Russian Federation)

armsid@mail.ru
ResearcherID B-5089-2017
ORCID 0000-0001-5312-7564

Anna V. Dolgova, Candidate of Technical Sciences, Assistant professor, Assistant professor of department "Computer Science and Computer graphics" (Omsk, Russian Federation)

9059238462@mail.ru
ResearcherID D-9708-2019
ORCID 0000-0002-8201-8823

Svetlana P. Zheleznyak, Candidate of Technical Sciences, Assistant professor of department "Computer Science and Computer graphics", Omsk State Transport University (Omsk, Russian Federation)

zhsp120866@yandex.ru
ResearcherID AAM-5444-2021
ORCID 0000-0001-8957-2678

УДК 519.682.7
DOI 10.25205/1818-7900-2021-19-2-115-130

Формирование композиций сервисов на основе статистических данных пользователей

Р. К. Фёдоров, И. В. Бычков, Г. М. Ружников

Институт динамики систем
и теории управления имени В. М. Матросова СО РАН
Иркутск, Россия

Аннотация

Рассмотрено решение задачи автоматического формирования композиций сервисов. Предложенный метод формирует композиции сервисов на основе обработки статистических данных об отдельных применениях сервисов (заданий) пользователями. В основе метода лежит определение связей вызовов сервисов друг с другом по данным. Далее выделяются параметры сервисов, значения которых жестко заданы композицией сервисов, и параметры, значения которых может изменить пользователь. Формируются композиции сервисов в виде направленного графа заданий DAG. Разработаны методы редукции множества получаемых композиций сервисов, позволяющие выделить завершенные и ранжировать их по степени используемости. В частности, определяются эквивалентные композиции сервисов на основе изоморфизма графов DAG, отбрасываются тривиальные и оставляются только композиции, приводящие к публикуемому результату.

Ключевые слова

SOA, OGC, WPS, обработка пространственных данных, семантическая сеть, композиция сервисов

Благодарности

Работа выполнена при поддержке ЦКП ИИВС ИРНОК, базового проекта АААА-А17-117032210079-1, гранта Минобрнауки России № 075-15-2020-787, проектов РФФИ 18-07-00758, 17-57-44006, блока интеграционной программы ИНЦ СО РАН 4.1.2

Для цитирования

Фёдоров Р. К., Бычков И. В., Ружников Г. М. Формирование композиций сервисов на основе статистических данных пользователей // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 115–130. DOI 10.25205/1818-7900-2021-19-2-115-130

Building Service Compositions Based on data on Use of Services by Users

R. K. Fedorov, I. V. Bychkov, G. M. Rugnikov

Matrosov Institute for System Dynamics and Control Theory SB RAS
Irkutsk, Russian Federation

Abstract

The automatic service composition is discussed in the article. The method is proposed for building the service composition based on the processing of statistical data on individual applying services (tasks) by users. The method is based on linking tasks to each other, determining data dependencies, parameters of services whose values are rigidly set by the composition of services, and parameters whose values can be changed by the user are highlighted. Service compositions are built in the form of a directed graph of DAG. The methods have been developed for reducing the set of obtained service compositions, which allow us to highlight useful ones and rank them by degree of use. In particular, equivalent service compositions based on isomorphism of DAG graphs are determined, trivial ones are discarded, and only compositions that lead to the published result are left behind.

Keywords

SOA, OGC, WPS, spatial data processing, semantic network, composition of services

Acknowledgements

This work was supported by the Center for Collective Use of the IISS IRNOK, no. AAAA-A17-117032210079-1, no. 075-15-2020-787 of the Ministry of Science and Higher Education of the Russian Federation, no. 0348-2018-0003 of the SB RAS, RFBR Projects no. 18-07-00758, 17-57-44006, ISC SB RAS integration program 4.1.2

For citation

Fedorov R. K., Bychkov I. V., Rugnikov G. M. Building Service Compositions Based on data on Use of Services by Users. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 115–130. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-115-130

Введение

В современном мире активно растет количество Web-сервисов [1], выполняющих предоставление, обработку и публикацию данных, в частности сервисы для обработки и предоставления данных дистанционного зондирования земли (ДЗЗ). Широко используются облачные хранилища данных, такие как интернет-сервисы: Dropbox, OneDrive, Google Drive, iCloud, Яндекс.Диск, Облако Mail.Ru и др. Применяются различные стандарты и спецификации, унифицирующие вызов сервисов, передачу и получение данных, например стандарт обработки пространственных данных OGC WPS [2], набор спецификаций REST [3]. Формируются каталоги сервисов, содержащих метаинформацию, для поиска необходимых сервисов. Сервисы активно применяются в решении сложных информационно-вычислительных задач, упрощая использование программного обеспечения за счет исключения процессов его установки и настройки.

Обычно для пользователей, не являющихся специалистами в области информационных технологий, применение сервисов реализовано с помощью специальной формы, где можно указать значения параметров и запустить сервис. Часто решение задачи может требовать многократного вызова последовательности (цепочки) сервисов, при этом пользователю приходится придерживаться определенного порядка вызова сервисов, вводить в форме значения параметров для каждого сервиса и т. д. В этом случае для автоматизации работы пользователя требуется создание композиции Web-сервисов [4]. Обычно композиции сервисов задаются с помощью языков для формального описания бизнес-процессов, например BPEL (Business Process Execution Language), BPMN (Business Process Modeling Notation), BPML (Business Process Management Language) и XPDL (XML Process Definition Language). Также для задания композиций сервисов активно используются популярные языки программирования, например Python, JavaScript. Существует ряд систем workflow («поток задач»), позволяющих формировать композиции сервисов в различных предметных областях: Pegasus [5], Kepler [6], Swift [7], KNIME [8], Taverna [9], Galaxy [10], Trident [11] and Triana [12], Everest (Mathcloud) [13], CLAVIRE [14]. Для обработки пространственных данных активно используется Geo-Processing Workflows [15]. Общепризнанной формализацией композиции сервисов является направленный ациклический граф (Directed acyclic graph, DAG) [16; 17], в котором вершинами являются вызовы сервисов, а дугами – зависимости между сервисами по данным. Создание композиции сервисов – это сложный процесс, требующий навыков программирования от пользователя, поиска подходящих сервисов среди их большого количества, знания языков задания композиций сервисов, больших временных издержек на разработку и отладку. Существует несколько подходов к автоматизации создания композиций сервисов [18].

1. Интерактивное построение композиций сервисов на основе знаний. В этом подходе пользователь вручную строит композицию сервисов, а система вывода и знания (метаописания, онтологии) используется для поиска подходящих сервисов и значений параметров [19–21].

2. Автоматическое построение композиций сервисов. В этом подходе производится (полу) автоматическое обнаружение и составление необходимых сервисов на основе семантического анализа и логического вывода [22–25].

3. Автоматическая композиция Web-сервисов с учетом планирования выполнения. В этом подходе [26; 27] входные данные представляются сервисами, которые семантически аннотированы с использованием онтологий, например Web Ontology Language for Services (OWL-S) [28], сервисы обработки аннотированы с помощью Web Service Modeling Ontology (WSMO) [29]. Выбор среди сгенерированных цепочек сервисов может проводиться на основе оценки качества обслуживания (QoS) с использованием алгоритмов, таких как генетические алгоритмы или алгоритмы теории игр [30; 31].

Следует отметить, что автоматическое построение композиций сервисов на основе семантического анализа, логического вывода и планирования является сложной задачей по следующим причинам:

- большая часть сервисов имеет недостаточно метаинформации, либо она отсутствует или не формализована;
- требуется проработанная онтология предметной области, ее построение не тривиальный процесс, в котором нужно рассмотреть все возможные связи между сервисами;
- требуется согласованность метаинформации сервисов и онтологии предметной области.

Обычно при решении конкретной задачи пользователь составляет некоторую последовательность выполнения сервисов на основе их неформализованных описаний и примеров решений подобных задач. Одна и та же последовательность выполнения сервисов может повторяться многократно. Автоматизация выполнения последовательности сервисов позволит упростить и ускорить работу пользователя. Выполняя последовательность сервисов, пользователь задает неявную информацию о связях между сервисами, используемых значениях параметров сервисов, которая может быть полезна для автоматического составления композиций сервисов. Поэтому актуальна задача автоматизации формирования композиций сервисов на основе статистических данных о применении сервисов пользователями. Авторами предлагается метод, который связывает одиночные вызовы сервисов друг с другом и определяет зависимости по данным. В статье будем рассматривать анализ только статистических данных и намеренно опускаем возможность применения метаданных и онтологий, оставляя их для дальнейшего исследования.

Сбор статистических данных о применении сервисов

В рамках геопортала ИДСТУ СО РАН [32] создана и развивается распределенная информационно-вычислительная среда, которая в настоящее время применяется преимущественно для задач обработки пространственных данных. Разработаны информационно-вычислительные сервисы, реализованные в соответствии со стандартом WPS (OGC, Open Geospatial Consortium) и сервисы хранения данных, каталог WPS-сервисов и системы выполнения сервисов и их композиций. Реализовано взаимодействие между WPS-сервисами и сервисами хранения данных, что позволяет упростить использование сервисов пользователем. Для каждого сервиса известно его расположение, список входных и выходных параметров, типы данных. При выполнении сервисов и их композиций автоматически формируются данные о произведенных вызовах сервисов, которые включают название сервиса и его адрес, значения входных и выходных параметров, время выполнения сервисов, успешность выполнения, ошибки выполнения и т. д. Данные сохраняются в таблице, где значения параметров хранятся в формате JSON (см. далее). Статистические данные о применении сервисов используются для формирования рекомендаций сервисов для пользователей [33; 34].

Таблица заданий
Task table

Идентификатор задания	Идентификатор сервиса	Статус	Входные параметры	Выходные параметры	Начало	Конец
3986	309	Success	{"table":"ticki"}	{"file":"http://84	18:20	18:22
3989	307	Failed	{"disp":"20"}		18:21	
3985	308	Success	{"file":"http://84..."}	{"res":"http://84	18:21	18:22
3987	307	Success	{"file":"http://84..."}	{"dist":"28",	19:18	19:19

Введем следующие обозначения, необходимые для описания метода:

$s=<name, I, O>$ – сервис, где *name* – это имя сервиса, I – множество входных параметров, O – множество выходных параметров сервиса. Далее будем обозначать $s.I$ и $s.O$ параметры, принадлежащие определенному сервису. Пользовательский вызов сервиса имеет одинаковую структуру: исходные параметры, известные пользователю, и параметры, которые должны быть получены;

$t=<s,V_I,V_O>$ – задание, т. е. вызов сервиса s с множеством значений V_I входных параметров $s.I$ и результаты V_O выходных параметров $s.O$;

Log – это множество заданий (успешных вызовов сервисов), совершенных пользователями. Не все вызовы сервисов успешные. Причиной неудачного вызова сервиса могут быть некорректные значения параметров, поэтому данные фильтруются, и оставляются только успешные вызовы.

На основе данных *Log* необходимо сформировать множество композиций сервисов в виде *DAG*.

Определение передачи данных между двумя заданиями

Первым этапом построения композиции сервисов является фиксация факта передачи данных между заданиями, которая производится на основе анализа данных, передаваемых в качестве значений параметров. Параметры сервисов в общем можно разделить на строковые, числовые и идентификаторы ресурсов URI. Предполагается, что если значение выходного параметра совпадает со значением входного параметра, то между заданиями может быть факт передачи данных. При этом необходимо учитывать, что равенство значений входных и выходных параметров может быть случайным. Однако вероятность ввода пользователем значения входного параметра совпадающего со значением выходного параметра другого задания обычно не высока, но и не нулевая. Определение факта передачи данных между заданиями в общем случае сводится к задаче распознавания, которую планируется исследовать в будущем. На текущий момент рассматриваются только идентификаторы ресурсов URI, не требующие применения методов распознавания. Обычно URI – это ссылки на файлы, сервисы предоставления данных и т. д., которые однозначно идентифицируют данные. Например, в современных распределенных базах данных в качестве первичного ключа вместе GUID часто используют URI. Большинство сервисов для каждого результата обработки генерируют уникальный URI. Это необходимо, чтобы гарантировать передачу результата, соответствующего запросу, в условиях часто одновременной обработки данных для множества клиентов сервиса. Следовательно, по совпадению URI значений параметров можно однозначно определить, что данные передаются от одного задания другому, т. е. если сервис однажды произвел данные, которые были использованы другим сервисом, то эти сервисы точно могут быть связаны по данным (достаточное условие связанности). Со строковыми и числовыми параметрами такой критерий не сработает, их можно будет использовать только как «необходи-

мое условие» связывания сервисов. Строковые и числовые параметры планируется рассмотреть в будущем.

Приведем алгоритм проверки передачи данных между двумя заданиями.

Input: *ti*, *tj* – два задания

Output: flag – наличие связи между заданиями.

```
1   function islinked(t_i, t_j)
2   flag <- False
3   for each v_m in t_i.VO:
4     for each v_n in t_j.VI:
5       If (typeof v_m = URI and typeof v_n = URI and v_m =v_n) then
6               flag <- true
7         v_n.linkedwith <- v_m
8       end if
9     end for
10  end for
11  return flag
```

Алгоритм проводит сравнение значений выходных параметров одного задания со значениями входных параметров другого задания, которые являются ссылками на файлы (URI). При совпадении значений URI параметров фиксируется наличие связи между двумя заданиями и определяется, по каким параметрам они связаны. Далее эти параметры будем называть связанными, а остальные параметры свободными.

Создание направленного ациклического графа заданий

Применение алгоритма проверки передачи данных для каждой пары заданий приводит к созданию направленного ациклического графа DAG_{gen} (рис. 1), где вершины задания t_i, а направленные дуги обозначают передачу данных между заданиями. Дополнительно получаем значения параметров заданий, введенных пользователем.

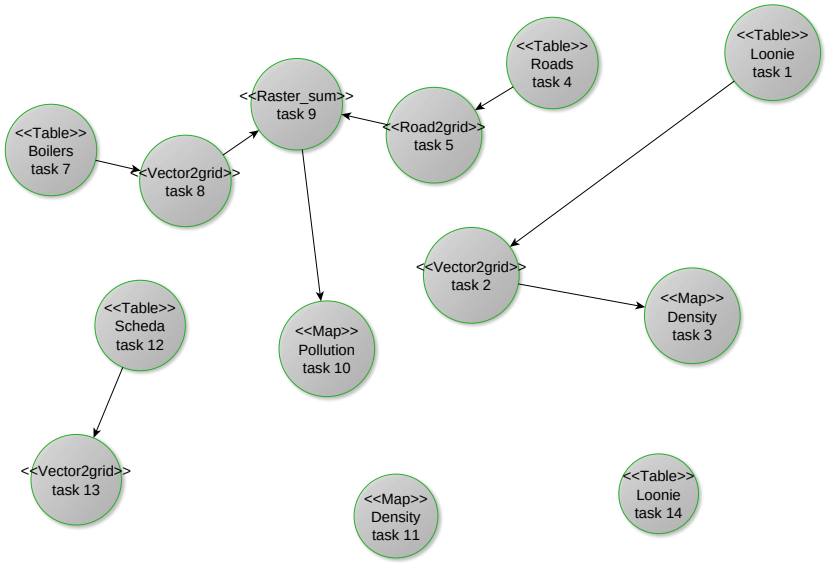


Рис. 1. Граф заданий DAG_{gen}
Fig. 1. The graph of tasks DAG_{gen}

Если при выполнении данного алгоритма сохранять связи между сервисами вместо заданий, то получим семантическую сеть сервисов. На рис. 2 представлена часть полученной сети. Сервисы представлены окружностями с подписанными названиями сервисов, семантические связи – направленными дугами. Необходимо отметить, что при таком подходе семантические связи между сервисами подтверждаются рабочими примерами.

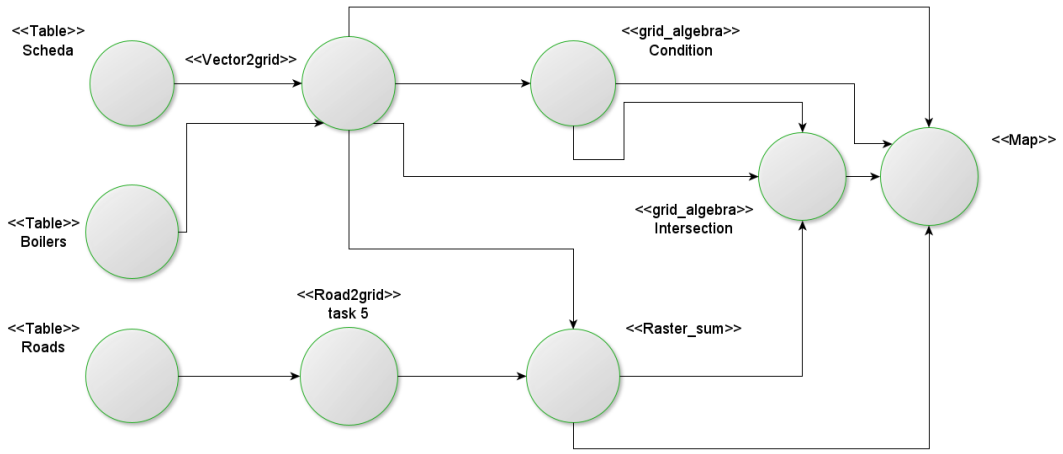


Рис. 2. Полученная семантическая сеть
Fig. 2. The resulting semantic network

Декомпозиция графа заданий

Граф заданий DAG_{gen} содержит результаты решения пользователем разных задач. Следовательно, его необходимо декомпозировать и выделить задания, соответствующие решаемым задачам.

Пользователь может выполнить некоторую последовательность сервисов, задающую последовательность заданий. Если задания связаны по данным (на текущий момент учитываются связи по URI), то они образуют связанный подграф D_i (компонент связности графа) в DAG_{gen}. Почти всегда выполнение композиции сервисов приводит к созданию такого связанного подграфа D_i. Если задания, сформированные композицией сервисов, не связаны, тогда такую композицию сервисов можно разделить. Для получения подграфа, соответствующего решению задачи, не достаточно найти компоненты связанности в DAG_{gen}. Так как в подграфе связанности могут быть задания получения данных. Например, задания могут выполняться сервисами получения данных ДЗЗ или сервисами предоставления векторных и реляционных данных. Эти данные могут использоваться для решения большого количества различных задач. Например, данные ДЗЗ Landsat 8, которые можно получить с помощью сервисов, используются для задач оценки подтоплений разливов рек, динамики изменения ледников, классификации породного состава лесов и т. д. Каждому решению задачи будет соответствовать свой подграф в DAG_{gen}, и они будут все связаны между собой через задание получения данных ДЗЗ, т. е. объединяться в компоненты связанности. Поэтому при выделении компонентов связности необходимо разделять подграфы, связанные только через задания получения данных. Учет и выделение заданий, выполняемых сервисами предоставления данных, при выполнении анализа может быть полезно для определения параметров композиций сервисов, т. е. сервисов, которые могут быть заменены на другие при ее повторном выполнении.

Для поиска связанных подграфов D_i применяется алгоритм поиска в ширину. При этом сложность поиска линейно зависит от суммы числа вершин и числа рёбер графа DAG_{gen}.

Алгоритм поиска связанного подграфа композиции сервисов

В статье для описания алгоритмов применяется псевдокод, использующий ключевые слова императивных языков программирования. Алгоритм поиска связанного подграфа реализует алгоритм поиска в ширину.

Input: nodes – список заданий, у каждого задания есть список предшествующих prevs и последующих заданий nects.

Output: subgraphs – список связанных подграфов.

```
1   Stack <-[];
2   while(nodes is not empty):// перебираем все задания
3         new subgraph
4         add subgraph into subgraphs
5         first <- nodes.pop()
6         add first into subgraph
7         add first into stack
8         while(stack is not empty){//перебор всех связанных заданий
9               cur <- stack.pop()
10              for each n in cur.nexts:
11                    if n exists in nodes then
12                          add n into subgraph.nodes
13                          add n into stack
14                          remove n in nodes
15                    end if
16              end for
17              for each n in cur.prevs:
18                    if n exists in nodes then
19                          add n into subgraph.nodes
20                          add n into stack
21                          remove n in nodes
22                    end if
23              end for
24        end while
25  end while
```

В результате выполнения алгоритма получаем множество связанных подграфов D_i (рис. 3). На рисунке задания получения данных выделены пунктирной линией. Задания публикации данных имеют более темную заливку.

Методы редукции множества композиций сервисов

Алгоритм поиска связанных подграфов D_i может получить их достаточно много. Часть подграфов может соответствовать экспериментам, которые привели к недостаточно хорошим результатам. Также среди них может быть достаточно много дублирований, которые возникают из-за многократного решения задачи на разных данных. Поэтому возникает задача редуцирования множества D_i. Композиции сервисов, состоящие из одной вершины (одного сервиса), являются тривиальными, и их можно отбросить. Одним из объективных критериев необходимости связанного подграфа заданий может быть получение пользователем конечного результата. Предполагается, что результат будет конечным, если пользователь его публикует. Например, на основе полученных данных создаются карты, графики, таблицы и т. д. В общем среди всех связанных подграфов выделяются D_i, состоящие как минимум из двух вершин и имеющие хотя бы одно задание публикации данных.

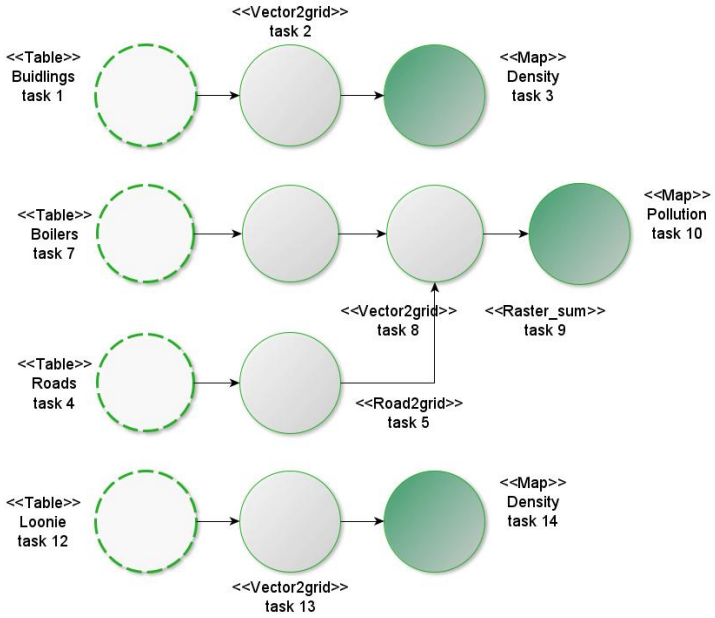


Рис. 3. Связанные подграфы D_i
Fig. 3. Linked subgraphs of D_i

Для декомпозиции графа заданий требуется предварительная классификация сервисов, которая производится вручную при регистрации сервиса в каталоге. Выделим три класса сервисов:

- предоставления данных, как правило, с них начинается выполнение композиции сервисов, в композиции сервисов они могут быть заменены;
- публикация данных, все сервисы данного класса должны требовать от пользователя метки семантики;
- вычислительные сервисы, выполняющие обработку данных.

Поиск эквивалентных подграфов

Часто последовательность применения сервисов многократно повторяются на разных наборах данных. При этом могут меняться входные данные, часть которых предоставляется сервисами. Это приводит к созданию нескольких подграфов D_i, в которых производятся вызовы одних и тех же вычислительных сервисов, но отличаются значениями параметров и сервисами предоставления данных. Заменим в связанных подграфах D_i задания t на соответствующие сервисы s. Ребра оставим без изменений, и в новом графе SD_i они вместо заданий будут связывать сервисы. Повторение одной и той же последовательность вызовов сервисов приведет к созданию изоморфных подграфов SD_i относительно сервисов и связей по данным без учета сервисов предоставления данных. Проверка на изоморфность двух графов осуществляется ниже следующим алгоритмом.

Алгоритм выделения изоморфных графов

Графы SD_i являются направленными, что значительно упрощает их сравнение. Предварительно для каждого графа добавляется узел, с которого начинается выполнение композиции. Начальное задание не имеет реальный вызов сервиса. Все задания, не зависящие по данным

от других, добавляются к начальному заданию. Сравнение двух графов начинается с начальных заданий и выполняется рекурсивно. Два задания являются изоморфными, если названия сервисов совпадают, и являются изоморфными все зависящие от них задания.

Input: n1, n2 – графы SD_i.

Output: true или false, являются графы изоморфными или нет.

```
1   function isonodes(n1,n2)
2   if n1.nexts.length <> n2.nexts.length then
3     return false
4   end if
5   for each nx1 in n1.nexts:
6     found <- false
7     for each nx2 in n2.nexts:
8       if nx1.service = nx2.service and isonodes(nx1,nx2) then
9         found <- true
10        break
11      end if
12    end for
13    if not found then
14      return false
15    end if
16  end for
17  return true
```

Изоморфные подграфы SD_i не всегда соответствуют одной и той же композиции сервисов. Например, один и тот же набор сервисов может применяться для получения разных результатов при изменении входных параметров. При дешифрировании данных ДЗЗ могут использоваться параметры, задающие различные пороговые значения. Их изменения могут привести к получению принципиально других результатов. С другой стороны, изменение параметров не всегда означает применение другой композиции вычислительных сервисов. Например, композиция сервисов может быть применена для другой территории, т. е. изменен параметр, задающий территорию, но решает ту же самую задачу. Возникает проблема определения, соответствуют ли два изоморфных графа решению одной задачи. Одним из способов решения этой проблемы является определение соответствия семантики результатов изоморфных графов. Автоматическое определение семантики результатов является нетривиальной задачей, поэтому предлагается воспользоваться метками пользователей. Пользователи часто сами дают такие метки, например названия карт, таблиц и графиков. В рамках среды можно обязать пользователей при публикации данных вручную указывать эти метки.

На рис. 4 приводится результат определения двух изоморфных подграфов SD_i, которые имеют одинаковую метку «Density» на сервисе публикации карт «Map». Изоморфные подграфы SD_i с одинаковой семантикой результатов объединяются (рис. 5). Количество изоморфных подграфов SD_i является оценкой частоты использования этой композиции сервисов и может применяться для их ранжирования и выделения наиболее полезных композиций сервисов, например, как работе [33].

Выполнение композиций сервисов

Подграфы SD_i являются искомыми композициями сервисов. Для того чтобы выполнить композицию сервисов, представленную подграфом SD_i, необходимо пользователю задать все свободные входные параметры, и переопределить сервисы предоставления данных. Все эти данные могут быть введены пользователем на генерируемой форме выполнения композиции

сервисов. В качестве значений по умолчанию предлагаются наиболее часто используемые значения среди всех изоморфных подграфов SD_i.

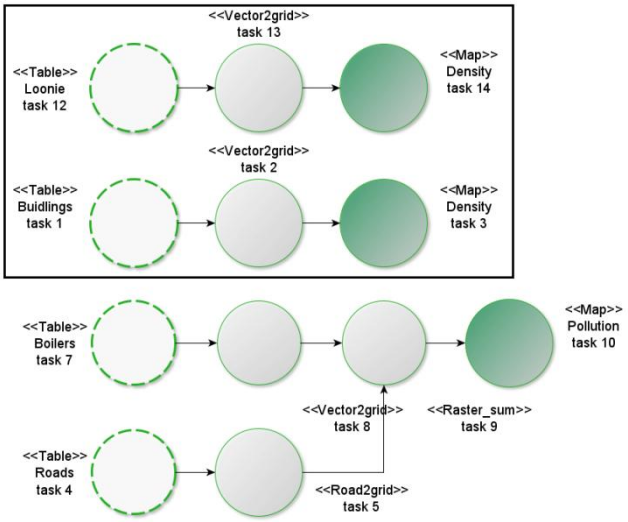


Рис. 4. Изоморфные подграфы выделены прямоугольником
Fig. 4. Isomorphic subgraphs are highlighted with a rectangle

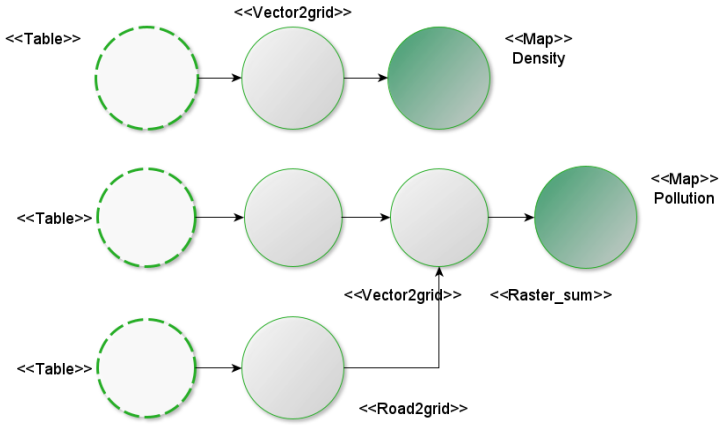


Рис. 5. Композиции сервисов
Fig. 5. Service compositions

Апробация

Для многих пространственных данных часто строят карты плотности (тепловые карты), которые дают представление о пространственном распределении объектов. Например, при изучении того или иного вида растения является важным исследование ареала его распространения. Специалистами собирается информация о находках различных видов растений в виде таблицы. Для создания карты распространения вида на основе этой таблицы в рамках геопортала используется три сервиса:

1) Table – сохраняет таблицу геопортала в векторный формат SHAPE, пользователю необходимо указать таблицу, файл результата;

2) Vector2grid – подсчитывает количество растений в ячейках регулярной сетки и сохраняет в формате GeoTIFF, пользователю необходимо указать размер ячейки, файл в формате SHAPE, файл результата;

3) Map – создает карту ареала распространения вида, пользователю необходимо указать файл в формате GeoTIFF, название карты, классы и стили отображения данных на карте.

Для сравнения эффективности автоматизации выполнения сервисов произведем подсчет условных действий пользователя. Будем учитывать количество действий поиска сервисов и ввод параметров. Первый сервис требует три действия, второй и третий – четыре. Итого имеем 11 действий пользователя для создания карты ареала распространения вида.

Выполнив последовательно все три сервиса, пользователь автоматически получает композицию сервисов (рис. 6).

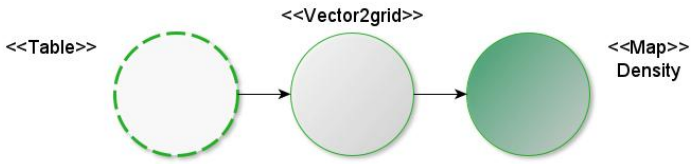


Рис. 6. Композиция сервисов для построения ареала распространения
Fig. 6. The service composition for calculating distribution area

При выполнении данной композиции пользователю необходимо выбрать композицию сервисов и указать таблицу, т. е. всего два действия пользователя. Конечно, любые свободные параметры сервисов могут быть изменены пользователем. В этом случае количество действий будет равно 5. Построение карты ареала распространения может выполняться для разных видов, по разным годам. Использование автоматически созданной композиции сервисов может значительно упростить работу пользователя.

Заключение

В статье рассмотрено решение актуальной задачи автоматического формирования композиций сервисов. Предложенный метод формирует композиции сервисов на основе обработки статистических данных об отдельных применениях сервисов (заданий) пользователями. В основе метода лежит определение связей вызовов сервисов друг с другом по данным. Далее выделяются параметры сервисов, значения которых жестко заданы композицией сервисов, и параметры, значения которых может изменить пользователь. Формируются композиции сервисов в виде направленного графа заданий DAG. Разработаны методы редукции множества получаемых композиций сервисов, позволяющие выделить полезные и ранжировать их по степени использования. В частности, определяются эквивалентные композиции сервисов на основе изоморфизма графов DAG, отбрасываются тривиальные и оставляются только композиции, приводящие к публикуемому результату.

В геопортале формируется список композиций сервисов, где у каждой композиции в качестве заголовка указывается название получаемых результатов. Список композиций сервисов ранжируется по мере их используемости, т. е. по количеству эквивалентных композиций сервисов. При выборе композиции пользователем формируется форма ввода параметров сервисов. Пользователю предлагается воспользоваться введенными значениями параметров сервисов или определить новые значения. После ввода параметров пользователь может выполнить композицию сервисов.

Разработанный метод имеет практическую значимость, которая заключается в том, что значительно упрощается разработка новых композиций для автоматизации повторяющихся действий пользователя. Для формирования композиции пользователю достаточно последовательно выполнить необходимые сервисы. Кроме того запоминаются значения параметров, что упрощает применение композиций сервисов. Кроме композиции сервисов строится их семантическая сеть, в которой семантические связи между сервисами подтверждаются работоспособным примером выполнения.

В будущем планируется развитие метода для реализации совместного использования метаданных, онтологий и статистических данных при построении композиций сервисов, распознавания семантики сервисов, разработки ранжирования на основе многопользовательских статистических данных.

Список литературы

1. **Grimm S., Abecker A., Volker J., Studer R.** Ontologies and the Semantic Web. *Handbook of semantic web technologies: foundations and technologies*, 2011, vol. 1, p. 507–579.
2. **Schut P.** OpenGIS ® Web Processing Service. *Open Geospatial Consortium*, 2007, no. 6, p. 1–3.
3. **Pautasso C.** RESTful Web service composition with BPEL for REST. *Data knowledge*, 2009, vol. 68, no. 9, p. 851–866.
4. **Hoffmann J., Weber I.** Web Service Composition. In: Encyclopedia of Social Network Analysis and Mining. Springer-Verlag, 2014.
5. **Deelman E., Vahi K., Juve G.** Pegasus, a workflow management system for science automation. *Future Generation Computer Systems*, 2015, vol. 46, p. 17–35.
6. **Ludäscher B., Altintas C., Berkley C., Higgins D., Jaeger E., Matthew J., Edward A. L., Tao J., Zhao Y.** Scientific Workflow Management and the Kepler System. *Special Issue: Workflow in Grid Systems. Concurrency and Computation: Practice & Experience*, 2006, vol. 18 (10), p. 1039–1065.
7. **Wilde M., Hategan M., Wozniak J. M.** Swift: A language for distributed parallel scripting. *Parallel Computing*, 2011, vol. 37 (9), p. 633–652.
8. **Berthold M. R., Cebron N., Dill F.** The konstanz information miner. *SIGKDD Explorations*, 2009, no. 11, p. 26–31.
9. **Wolstencroft K., Haines R., Fellows D.** The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. *Nucleic Acids Research*, 2013, vol. 41 (W1), p. 557–561.
10. **Blankenberg D., Kuster G. V., Coraor N.** Galaxy: A Web-Based Genome Analysis Tool for Experimentalists. Wiley, 2010.
11. **Simmhan Y., Barga R., Ingen C.** Building the trident scientific workflow workbench for data management in the cloud. In: Advanced Engineering Computing and Applications in Sciences (ADVCOMP), 2009. DOI 10.1109/ADVCOMP.2009.14
12. **Churches D., Gombas G., Harrison A.** Programming scientific and distributed workflow with Triana services: Research articles. *Concurrency and Computation: Practice and Experience*, 2006, vol. 18 (10), p. 1021–1037.
13. **Smirnov S., Sukhoroslov O., Volkov S.** Integration and Combined Use of Distributed Computing Resources with Everest. *Procedia Computer Science*, 2016, vol. 101, p. 359–368.
14. **Бухановский А. В., Васильев В. Н., Виноградов В. Н., Смирнов Д. Ю., Сухоруков С. А., Яппаров Т. Г.** CLAVIRE: Perspective Technology for Second Generation Cloud Computing // Современные тенденции развития распределенных вычислений. 2011, Т. 54, № 10. С. 7–14.

15. **Chen N. C., Di L. P., Yu G. N., Gong J. Y.** Geo-processing workflow driven wildfire hot pixel detection under sensor web environment. *Computers & geosciences*, 2010, vol. 36, no. 3, p. 362–372.
16. **Kwok Y.-K., Ahmad I.** Static scheduling algorithms for allocating directed task graphs to multiprocessors. *ACM Computing Surveys*, 1999, vol. 31, no. 4, p. 406–471.
17. **Xie G., Li R., Xiao X., Chen Y.** A High-Performance DAG Task Scheduling Algorithm for Heterogeneous Networked Embedded Systems. In: Proc. of IEEE 28th International Conference Advanced Information Networking and Applications, 2014, p. 1011–1016.
18. **Zhi-Wei H., Cheng-Zhi Q., A-Xing Z., Peng L., Yi-Jie W., Yun-Qiang Z.** From Manual to Intelligent: A Review of Input Data Preparation Methods for Geographic Modeling. *ISPRS International journal of geo-information*, 2019, vol. 8, no. 9, article 376. DOI 10.3390/ijgi8090376
19. **Di L., Zhao P., Yang W., Yue P.** Ontology-Driven Automatic Geospatial-Processing Modeling Based on Web-Service Chaining. In: Proceedings of the Sixth Annual NASA Earth Science Technology Conference. College Park, MD, USA, 2006, p. 27–29.
20. **Zhao P., Di L., Yu G., Yue P., Wei Y., Yang W.** Semantic Web-based geospatial knowledge transformation. *Computers & Geosciences*, 2009, no. 35, p. 798–808.
21. **Scheider S., Ballatore A.** Semantic typing of linked geoprocessing workflows. *International Journal of Digital Earth*, 2017, vol. 11, p. 113–138.
22. **Jiang J., Zhu A.X., Qin C.Z., Zhu T., Liu J., Du F., Liu J., Zhang G., An Y.** CyberSoLIM: A cyber platform for digital soil mapping. *Geoderma*, 2016, no. 263, p. 234–243.
23. **Lutz M., Lucchi R., Friis-Christensen A., Ostländer N.** A Rule-Based Description Framework for the Composition of Geographic Information Services. In: Proceedings of the International Conference on GeoSpatial Sematics. Mexico City, Mexico, 2007, p. 114–127.
24. **Lutz M.** Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *GeoInformatica*, 2007, vol. 11, p. 1–36.
25. **Lutz M., Lucchi R., Friis-Christensen A., Ostländer N.** A Rule-Based Description Framework for the Composition of Geographic Information Services. In: Proceedings of the International Conference on GeoSpatial Sematics. Mexico City, Mexico, 2007, p. 114–127.
26. **Yue P., Di L., Yang W., Yu G., Zhao P., Gong J.** Semantic Web Services-based process planning for earth science applications. *International Journal of Geographical Information Science*, 2009, vol. 23, p. 1139–1163.
27. **Farnaghi M., Mansourian A.** Automatic composition of WSMO based geospatial semantic web services using artificial intelligence planning. *Journal of Spatial Science*, 2013, vol. 58, p. 235–250.
28. **Martin D., Burstein M., Hobbs J., Lassila O., McDermott D., McIlraith S., Narayanan S., Paolucci M., Parsia B., Payne T.** OWL-S: Semantic markup for web services. W3C Member Submission, 2004.
29. **Roman D., Keller U., Lausen H., Bruijn J.D., Stollberg M., Polleres A., Feier C., Bussler C., Fensel D.** Web Service Modeling Ontology. *Applied ontology*, 2005, no. 1, p. 77–106.
30. **Li H., Zhu Q., Yang X., Xu L.** Geo-information processing service composition for concurrent tasks: A QoS-aware game theory approach. *Computers & Geosciences*, 2012, vol. 47, p. 46–59.
31. **Yue P., Tan Z., Zhang M.** GeoQoS: Delivering Quality of Services on the Geoprocessing Web. In: Proceedings of the OSGeo's European Conference on Free and Open Source Software for Geospatial (FOSS4G-Europe 2014). Bremen, Germany, 2014.
32. **Фёдоров Р. К., Бычков И. В., Шумилов А. С., Ружников Г. М.** Система планирования и выполнения композиций веб-сервисов в гетерогенной динамической среде // Вычислительные технологии. 2016. Т. 21, № 6. С. 18–35.
33. **Городничев М. А., Комиссаров А. В., Можина А. В., Прочкин П. В., Рудыч П. Д., Юрченко А. В.** Модели и проектные решения системы хранения и обработки исследо-

вательских данных Ecclesia // Вестник НГУ. Серия: Информационные технологии. 2018. Т. 16, № 3. С. 87–104.

34. **Linoff G. S., Berry M. J. A.** Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. 3rd ed. John Wiley & Sons, 2004. 643 p.

References

1. **Grimm S., Abecker A., Volker J., Studer R.** Ontologies and the Semantic Web. *Handbook of semantic web technologies: foundations and technologies*, 2011, vol. 1, p. 507–579.
2. **Schut P.** OpenGIS ® Web Processing Service. *Open Geospatial Consortium*, 2007, no. 6, p. 1–3.
3. **Pautasso C.** RESTful Web service composition with BPEL for REST. *Data knowledge*, 2009, vol. 68, no. 9, p. 851–866.
4. **Hoffmann J., Weber I.** Web Service Composition. In: Encyclopedia of Social Network Analysis and Mining. Springer-Verlag, 2014.
5. **Deelman E., Vahi K., Juve G.** Pegasus, a workflow management system for science automation. *Future Generation Computer Systems*, 2015, vol. 46, p. 17–35.
6. **Ludäscher B., Altintas C., Berkley C., Higgins D., Jaeger E., Matthew J., Edward A. L., Tao J., Zhao Y.** Scientific Workflow Management and the Kepler System. *Special Issue: Workflow in Grid Systems. Concurrency and Computation: Practice & Experience*, 2006, vol. 18 (10), p. 1039–1065.
7. **Wilde M., Hategan M., Wozniak J. M.** Swift: A language for distributed parallel scripting. *Parallel Computing*, 2011, vol. 37 (9), p. 633–652.
8. **Berthold M. R., Cebron N., Dill F.** The konstanz information miner. *SIGKDD Explorations*, 2009, no. 11, p. 26–31.
9. **Wolstencroft K., Haines R., Fellows D.** The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. *Nucleic Acids Research*, 2013, vol. 41 (W1), p. 557–561.
10. **Blankenberg D., Kuster G. V., Coraor N.** Galaxy: A Web-Based Genome Analysis Tool for Experimentalists. Wiley, 2010.
11. **Simmhan Y., Barga R., Ingen C.** Building the trident scientific workflow workbench for data management in the cloud. In: Advanced Engineering Computing and Applications in Sciences (ADVCOMP), 2009. DOI 10.1109/ADVCOMP.2009.14
12. **Churches D., Gombas G., Harrison A.** Programming scientific and distributed workflow with Triana services: Research articles. *Concurrency and Computation: Practice and Experience*, 2006, vol. 18 (10), p. 1021–1037.
13. **Smirnov S., Sukhoroslov O., Volkov S.** Integration and Combined Use of Distributed Computing Resources with Everest. *Procedia Computer Science*, 2016, vol. 101, p. 359–368.
14. **Boukhanovsky A. V., Vasilev V. N., Vinogradov V. N., Smirnov D. Y., Sukhorukov S. A., Yapparov T. G.** CLAVIRE: Perspective Technology for Second Generation Cloud Computing. *Priborostroenie*, 2011, vol. 54, no. 10, p. 7–14.
15. **Chen N. C., Di L. P., Yu G. N., Gong J. Y.** Geo-processing workflow driven wildfire hot pixel detection under sensor web environment. *Computers & geosciences*, 2010, vol. 36, no. 3, p. 362–372.
16. **Kwok Y.-K., Ahmad I.** Static scheduling algorithms for allocating directed task graphs to multiprocessors. *ACM Computing Surveys*, 1999, vol. 31, no. 4, p. 406–471.
17. **Xie G., Li R., Xiao X., Chen Y.** A High-Performance DAG Task Scheduling Algorithm for Heterogeneous Networked Embedded Systems. In: Proc. of IEEE 28th International Conference Advanced Information Networking and Applications, 2014, p. 1011–1016.
18. **Zhi-Wei H., Cheng-Zhi Q., A-Xing Z., Peng L., Yi-Jie W., Yun-Qiang Z.** From Manual to Intelligent: A Review of Input Data Preparation Methods for Geographic Modeling. *ISPRS In-*

ternational journal of geo-information, 2019, vol. 8, no. 9, article 376. DOI 10.3390/ijgi8090376

19. **Di L., Zhao P., Yang W., Yue P.** Ontology-Driven Automatic Geospatial-Processing Modeling Based on Web-Service Chaining. In: Proceedings of the Sixth Annual NASA Earth Science Technology Conference. College Park, MD, USA, 2006, p. 27–29.
20. **Zhao P., Di L., Yu G., Yue P., Wei Y., Yang W.** Semantic Web-based geospatial knowledge transformation. *Computers & Geosciences*, 2009, no. 35, p. 798–808.
21. **Scheider S., Ballatore A.** Semantic typing of linked geoprocessing workflows. *International Journal of Digital Earth*, 2017, vol. 11, p. 113–138.
22. **Jiang J., Zhu A.X., Qin C.Z., Zhu T., Liu J., Du F., Liu J., Zhang G., An Y.** CyberSoLIM: A cyber platform for digital soil mapping. *Geoderma*, 2016, no. 263, p. 234–243.
23. **Lutz M., Lucchi R., Friis-Christensen A., Ostländer N.** A Rule-Based Description Framework for the Composition of Geographic Information Services. In: Proceedings of the International Conference on GeoSpatial Sematics. Mexico City, Mexico, 2007, p. 114–127.
24. **Lutz M.** Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *GeoInformatica*, 2007, vol. 11, p. 1–36.
25. **Lutz M., Lucchi R., Friis-Christensen A., Ostländer N.** A Rule-Based Description Framework for the Composition of Geographic Information Services. In: Proceedings of the International Conference on GeoSpatial Sematics. Mexico City, Mexico, 2007, p. 114–127.
26. **Yue P., Di L., Yang W., Yu G., Zhao P., Gong J.** Semantic Web Services-based process planning for earth science applications. *International Journal of Geographical Information Science*, 2009, vol. 23, p. 1139–1163.
27. **Farnaghi M., Mansourian A.** Automatic composition of WSMO based geospatial semantic web services using artificial intelligence planning. *Journal of Spatial Science*, 2013, vol. 58, p. 235–250.
28. **Martin D., Burstein M., Hobbs J., Lassila O., McDermott D., McIlraith S., Narayanan S., Paolucci M., Parsia B., Payne T.** OWL-S: Semantic markup for web services. W3C Member Submission, 2004.
29. **Roman D., Keller U., Lausen H., Bruijn J.D., Stollberg M., Polleres A., Feier C., Bussler C., Fensel D.** Web Service Modeling Ontology. *Applied ontology*, 2005, no. 1, p. 77–106.
30. **Li H., Zhu Q., Yang X., Xu L.** Geo-information processing service composition for concurrent tasks: A QoS-aware game theory approach. *Computers & Geosciences*, 2012, vol. 47, p. 46–59.
31. **Yue P., Tan Z., Zhang M.** GeoQoS: Delivering Quality of Services on the Geoprocessing Web. In: Proceedings of the OSGeo's European Conference on Free and Open Source Software for Geospatial (FOSS4G-Europe 2014). Bremen, Germany, 2014.
32. **Fedorov R. K., Bychkov I. V., Shumilov A. S., Ruzhnikov G. M.** System for planning and executing web service compositions in a heterogeneous dynamic environment. *Computational technologies*, 2016, vol. 21, no. 6, p. 18–35.
33. **Gorodnichev M. A., Komissarov A. V., Mozhina A. V., Prochkin P. V., Rudych P. D., Yurchenko A. V.** Information Models and Project Solutions for the Ecclesia Research Data Storing and Processing System. *Vestnik NSU. Series: Information Technologies*, 2018, vol. 16, no. 3, p. 87–104. (in Russ.)
34. **Linoff G. S., Berry M. J. A.** Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. 3rd ed. John Wiley & Sons, 2004. 643 p.

Материал поступил в редколлегию
Received
23.04.2021

Сведения об авторах

Фёдоров Роман Константинович, кандидат технических наук, ведущий научный сотрудник, Институт динамики систем и теории управления имени В. М. Матросова СО РАН (Иркутск, Россия)

fedorov@icc.ru

Бычков Игорь Вячеславович, доктор технических наук, академик РАН, директор, Институт динамики систем и теории управления имени В. М. Матросова СО РАН (Иркутск, Россия)

bychkov@icc.ru

Ружников Геннадий Михайлович, доктор технических наук, зав. отделением, Институт динамики систем и теории управления имени В. М. Матросова СО РАН (Иркутск, Россия)

rugnikov@icc.ru

Information about the Authors

Roman K. Fedorov, Senior Researcher, V. M. Matrosov Institute of System Dynamics and Control Theory SB RAS (Irkutsk, Russian Federation)

fedorov@icc.ru

Igor V. Bychkov, Doctor of Technical Sciences, Academician of the Russian Academy of Sciences, Director, V. M. Matrosov Institute of System Dynamics and Control Theory SB RAS (Irkutsk, Russian Federation)

bychkov@icc.ru

Gennady M. Ruzhnikov, Doctor of Technical Sciences, Head of the Department, V. M. Matrosov Institute of System Dynamics and Control Theory SB RAS (Irkutsk, Russian Federation)

rugnikov@icc.ru

Правила оформления текста рукописи

Авторы представляют статьи на русском или английском языке объемом от 0,5 авторского листа (20 тыс. знаков) до 1 авторского листа (40 тыс. знаков), включая иллюстрации (1 иллюстрация форматом 190 × 270 мм = 1/6 авторского листа, или 6,7 тыс. знаков). Публикации, превышающие указанный объем, допускаются к рассмотрению только после индивидуального согласования с редакцией журнала.

Текст рукописи должен быть представлен в редколлегию в виде файла MS Word (.doc, .docx). Гарнитура Times New Roman, размер шрифта 11, межстрочный интервал 1, размеры полей – стандартные значения текстового редактора. Форматирование – выравнивание по ширине страницы, переносы слов включены, каждый новый абзац начинается с красной строки. Не допускается ручное форматирование абзацев (пробелами, лишними переводами строк, разрывами страниц).

Структура статьи

- Индекс УДК (универсальной десятичной классификации). Выравнивание по левому краю
- Название статьи. Выравнивание по центру, полужирный шрифт
- ФИО авторов (инициалы, фамилия). Выравнивание по центру, полужирный шрифт
- Места работы всех авторов. Выравнивание по центру, курсив
- Аннотация статьи
- Ключевые слова, не более 10
- Благодарности, сведения о финансовой поддержке
- Название статьи **на английском языке**. Выравнивание по центру, полужирный шрифт
- ФИО авторов **на английском языке** (инициалы, фамилия). Выравнивание по центру, полужирный шрифт
- Места работы авторов **на английском языке**. Выравнивание по центру, курсив
- Аннотация статьи **на английском языке (Abstract)**, 200–250 слов
- Ключевые слова **на английском языке (Keywords)**, не более 10
- Благодарности, сведения о финансовой поддержке **на английском языке**, если есть соответствующий раздел на русском языке **(Acknowledgements)**
- Основной текст
- Список литературы / **References**
- Сведения об авторах

Требования к оформлению основного текста и иллюстративных материалов

Основной текст должен быть представлен в структурированном виде, рекомендуется использовать подзаголовки – например: Введение, Методика..., Выводы, Результаты, Заключение.

Подзаголовки отделяются и набираются полужирным шрифтом. В целях выделения частей текста и отдельных слов и словосочетаний допускается использование курсива или полужирного шрифта. Подчеркивание, разрядка, изменение основного кегля и выделение цветом не используются.

Иллюстрации к рукописи статьи должны быть приложены в виде отдельных файлов. При этом в тексте должно содержаться включенное изображение с указанием имени файла. Все

иллюстрации, содержащие схемы, графики, алгоритмы и т. п., должны быть представлены в векторном виде (.ai, .eps, .cdr). Скриншоты и другие растровые изображения должны быть представлены в максимально высоком качестве, без каких-либо потерь и искажений (.jpg, .tif). Все иллюстрации должны иметь подрисуночную подпись – свое название. Надписи к таблицам и подписи к иллюстрациям приводятся **на двух языках (русском и английском).**

Примеры:

Рис. 1. Диаграмма производительности...
Fig. 1. Performance diagram...

Таблица 1

Сравнение алгоритмов...

Table 1

Comparison of algorithms...

Нумерация последовательная и неразрывная от начала статьи. Не допускается использование других наименований, кроме «Рис.» / «Fig.», «Таблица» / «Table», и усложнение нумерации (например, «Рис. 3.2.»). Ссылка на иллюстрацию в тексте должна быть приведена в круглых скобках, например: (рис. 1), (табл. 1).

Формулы должны быть набраны с использованием редактора MathType либо встроенного редактора формул MS Word. Кегль основных символов – 11, греческие символы набираются прямым шрифтом, латинские – курсивом. Нумеруются только те формулы, на которые автор ссылается в тексте.

Abstract

Аннотация статьи на английском языке (Abstract) не должна быть дословным переводом русскоязычной аннотации. Раздел Abstract, как и основной текст, должен быть структурирован, в нем должно содержаться описание цели работы, методов исследования, научной значимости, выводов / результатов. Требуется качественный перевод на английский язык (при необходимости просим авторов обращаться к профессиональным переводчикам). **Объем Abstract 200–250 слов.**

Список литературы / References

Список литературы и список литературы на английском языке (References) размещаются в общем разделе. Рекомендуемое количество цитируемых в статье источников – не менее 10, в список желательно включать ссылки на актуальные работы по теме исследования, особенно в иностранных периодических изданиях.

В тексте статьи ссылки на литературу указываются цифрами в квадратных скобках, при необходимости указываются номера страниц, например: [2; 3. С. 15].

Список литературы нумеруется в порядке цитирования и оформляется в соответствии с ГОСТ Р 7.0.5-2008 на библиографическое описание (знаки тире в описании опускаются). Ссылки на неопубликованные работы, а также на Интернет-ресурсы (кроме электронных изданий, поддающихся библиографическому описанию) оформляются в виде сноски.

В Список литературы ссылки на источники следует включать на оригинальном языке опубликования. Каждый источник должен быть также оформлен на английском языке (References) по международному стандарту для публикаций в области информатики IEEE Style со следующими отличиями:

- инициалы авторов указываются после фамилии;
- название статьи не берется в кавычки, отделяется точкой;
- отсутствует союз «and» перед фамилией последнего автора;
- в диапазоне страниц указывается одна «p» (вместо «pp. 2–9» – «p. 2–9»);
- год издания указывается после места издания (для книг) и сразу после названия журнала (для периодики).

Перевод источника на английский язык:

- если источник имеет выходные данные на английском языке, то для формирования References **следует использовать именно эти данные**;
- если оригинальная публикация не содержит выходных данных на английском языке, то допускается транслитерация названия материала на латинский алфавит в сочетании с переводом на английский язык в квадратных скобках. В конце описания указывается, на каком языке написана эта работа, например, (in Russ.). При транслитерации можно воспользоваться Интернет-ресурсом http://ru.translit.ru/, рекомендуется выбрать стандарт BSI. Место издания не транслитерируется, указывается полностью на английском языке, например: Moscow. Название издательства / издателя, как правило, транслитерируется. Для журналов, у которых есть официальное название на английском языке, – использовать его (проверить на сайте журнала, или, например, в библиотеке WorldCat), если названия на английском языке нет, использовать транслитерацию по системе BSI. Не следует самостоятельно переводить названия журналов.

Если у цитируемого источника есть **цифровой идентификатор DOI** (https://search.crossref.org), его требуется обязательно указывать в конце библиографической ссылки.

Примеры оформления ссылок. Каждый источник в том же пункте дублируется на английском языке (References).

Источник на русском языке, перевод на английский доступен в метаданных статьи

1. **Журавлев С. С., Рудометов С. В., Окольнишников В. В., Шакиров С. Р.** Применение модельно-ориентированного проектирования к созданию АСУ ТП опасных промышленных объектов // Вестник НГУ. Серия: Информационные технологии. 2018. Т. 16, № 4. С. 56–67. DOI 10.25205/1818-7900-2018-16-4-56-67

Zhuravlev S. S., Rudometov S. V., Okolnishnikov V. V., Shakirov S. R. Model-Based Design Approach for Development Process Control Systems of Hazardous Industrial Facilities. *Vestnik NSU. Series: Information Technologies*, 2018, vol. 16, no. 4, p. 56–67. (in Russ.) DOI 10.25205/1818-7900-2018-16-4-56-67

Источник на английском языке. Оформляем согласно требованиям для References. Приводим только 1 раз.

2. **Telnov V. I.** Optimization of the Beam Crossing Angle at the ILC for E + e- and yy Collisions. *Journal of Instrumentation*, 2018, vol. 13, no. 03, p. P03020–P03020. DOI 10.1088/1748-0221/13/03/p03020

Метаданные источника доступны только на русском языке

3. **Жижимов О. Л., Федотов А. М., Шокин Ю. И.** Технологическая платформа массовой интеграции гетерогенных данных // Вестник НГУ. Серия: Информационные технологии. 2013. Т. 11, вып. 1. С. 24–41ю

Zhizhimov O. L., Fedotov A. M., Shokin Yu. I. Tekhnologicheskaya platforma massovoi integratsii geterogennykh dannykh [Technology Platform For the Mass Integration of Heterogeneous Data]. *Vestnik NSU. Series: Information Technologies*, 2013, vol. 11, no. 1, p. 24–41. (in Russ.)

Сведения об авторах

Последний раздел статьи – информация об авторе / авторах **на русском и английском языках**:

- ФИО полностью, ученая степень, ученое звание, должность, место работы, адрес места работы
- e-mail
- идентификаторы автора, такие как ORCID или ResearcherID (всем авторам рекомендуется использовать данные сервисы для ведения актуального списка своих публикаций)
- контактный телефон (не публикуется)

Если статья представляется на английском языке, необходимо приложить перевод на русский язык названия, аннотации, ключевых слов, сведений об авторе.

Доставка материалов

Материалы предоставляются в редакцию по электронной почте inftech@vestnik.nsu.ru.

Порядок рецензирования

Все статьи сначала проходят проверку на заимствование и только после этого отправляются на рецензирование. Редакционный совет не допускает к публикации материал, если имеется достаточно оснований полагать, что он является плагиатом.

Тип рецензирования статей – двухуровневое, одностороннее анонимное («слепое»).

Для каждой статьи редколлегией выбираются рецензенты, научная деятельность которых связана с темой представленного материала. Ответственный секретарь журнала обращается к ним с просьбой дать экспертную оценку статье либо помочь организовать рецензирование.

Рецензии для журнала «Вестник НГУ. Серия: Информационные технологии» составляются по единой схеме и подразумевают оценку по следующим критериям: соответствие тематике журнала, оригинальность и значимость результатов, качество изложения материала.

Заполненный бланк рецензии высылается на электронный адрес редакции. В зависимости от экспертных заключений статья может быть принята редакционным советом к опубликованию, рекомендована автору к доработке (с последующим повторным рецензированием либо без него) или отклонена (с предоставлением автору мотивированного отказа). Автору на электронный адрес высылается текст рецензии без указания ФИО рецензента и его контактных данных.

Все рецензии хранятся в редакции журнала не менее 5 лет. Редколлегия журнала обязуется при поступлении соответствующего запроса направлять копии рецензий в Министерство науки и высшего образования Российской Федерации.