ОГЛАВЛЕНИЕ

УДК 004.5; 004.657

ИСПОЛЬЗОВАНИЕ СЕМАНТИЧЕСКИХ СВЯЗЕЙ ОНТОЛОГИИ ДЛЯ СОЗДАНИЯ АДАПТИВНОГО ИНТЕРФЕЙСА

О. М. Атаева[1] [0000-0003-0367-5575], В. А. Серебряков[2] [0000-0003-1423-621X], Н. П. Тучкова[3] [0000-0001-5357-9640]

[1,2,3]*Вычислительный центр им. А.А. Дородницына ФИЦ Информатика и управление РАН, г. Москва*

[1]oli@ultimeta.ru, [2]serebr@ultimeta.ru, [3]natalia_tuchkova@mail.ru

Аннотация

Работа посвящена проблеме настройки пользовательских интерфейсов информационной системы, осуществляющей интеграцию данных. Настраиваемый интерфейс служит одним из средств организации представления данных предметной области. Изучен вопрос об использовании семантических связей онтологии для подбора данных, соответствующих задачам исследований. Рассмотрена модель адаптивного интерфейса, который позволяет наиболее точно отразить потребности исследователя в рамках определенной предметной области. Показано, как средствами, заложенными в модели семантической библиотеки, формируется адаптивный интерфейс.

***Ключевые слова**: онтология, адаптивный интерфейс, предметная область, модель данных*

1. ВВЕДЕНИЕ

Адаптивными информационными системами [1] (АИС) называют системы, в которых заложены алгоритмы, изменяющие поведение системы в ответ на действия пользователей или обладающие возможностью модифицировать свой контент не только на уровне данных, но и на уровне связей данных и, соответственно, модифицировать их представление пользователю. В рамках настоящей работы ограничимся рассмотрением интерфейса, настраиваемого в результате действия пользователя. Настройка интерфейса происходит на основе алгоритмов, заложенных в семантической библиотеке как следствие работы пользователя в опре-

деленной предметной области (ПрО) интегрированной системы данных цифровой библиотеки. Адаптация системы на уровне настраиваемого интерфейса направлена на обеспечение информационной потребности пользователя при уменьшении информационного шума.

Таким образом, рассматривается подход к проектированию семантической цифровой научной библиотеки, в которой реализуются механизмы формирования контента «под конкретного пользователя». Благодаря такому подходу осуществляется более углубленный поиск в ПрО пользователя. При этом ПрО пользователя – это то, что предоставляет семантическая библиотека на основе поисковых запросов пользователя, то есть это «умная выборка» из контента библиотеки.

Идея адаптации интерфейсов информационных систем возникла естественным образом у многих разработчиков на этапе интеграции данных и необходимости обеспечить пользователя специфическим интерфейсом, характерным для определенных задач [1–3]. В процессе накопления данных в семантической библиотеке устанавливается множество связей. Связи необходимы для обеспечения полной информационной поддержки пользователей на базе контента информационной системы. Тем не менее, на определенном этапе пользователь может столкнуться с «лишней» информацией, которая не соответствует его интересу, т. е. пертинентностью ответа системы. В этом случае система должна адекватно реагировать на запросы, аккумулируя информацию о пользователе, что и происходит в современных поисковых ресурсах. Развитие этого подхода привело к переходу от «монолитных», единых сервисов информационных систем к «микросервисам» [4–6].

В отношении научных знаний, представляемых в цифровых семантических библиотеках, адаптация интерфейсов системы связана с предоставлением пользователю информации из определенной ПрО и интегрированных данных [7–9]. Если знания связаны с другими научными областями, приложениями, авторами, публикациями, то естественно, что информационная система должна «очертить область интересов» и предоставить пользователю возможность выбора информации при поиске.

В предлагаемой работе рассмотрена адаптация интерфейса семантической

библиотеки LibMeta[1] и ее математического контента, который опирается на классические источники академического сообщества, такие как Математическая энциклопедия, классификаторы MSC[2] и УДК[3], авторские тезаурусы[4] и словари и др.

2. АИС СЕМАНТИЧЕСКИХ БИБЛИОТЕК

Выделяют несколько видов АИС [1–3].

- Системы, которые *в зависимости от действий пользователя* отображают информацию, соответствующую его информационным потребностям. Это может достигаться за счет сложной организации ключевых слов, графических объектов, процессов, в которые вовлечен пользователь в информационной системе, и т. д. Такие системы адаптируются под действия пользователя, варьируя как способ представления информации, так и саму информацию (например, разные пользователи перемещаются по списку найденной информации в разном порядке, и это в дальнейшем влияет на то, какая информация будет им прежде всего отображаться, или отслеживается последовательность поисковых запросов, чтобы предложить контекстную рекламу). В таких системах *модель данных жестко структурирована*.
- Обучающиеся системы, в основе которых лежат *методы автоматической классификации контента*, благодаря которым выстраивается определенная сеть иерархических и горизонтальных связей в поступающей информации. Система адаптируется под контент, а способы представления информации и сама представляемая информация не зависят от внешних условий (например, действий пользователя). Отличительной особенностью таких систем является *возможность изменять модель данных*, что отражается на классификации и представлении данных в системе.

Семантические библиотеки [10–13] относятся ко второму типу систем. С одной стороны, в таких библиотеках возникает необходимость ограничить контент в рамках некоторой ПрО. Для этого используется набор терминов, описывающих эту ПрО. Чаще всего эти термины организованы в виде некоторого тезауруса. С

[1] https://libmeta.ru/
[2] MSC2020-Mathematics Subject Classification System. https://zbmath.org/static/msc2020.pdf
[3] https://teacode.com/online/udc/
[4] ГОСТ 7.24-2007 Тезаурус информационно-поисковый многоязычный. https://ifap.ru/library/gost/7242007.pdf

другой стороны, наполнение библиотеки представляет собой множество публикаций, книг, проектов, задач, т. е. это разные ресурсы, перечень которых может изменяться. Изменяются также структура и связи этих ресурсов. Для тематической классификации ресурсов библиотеки могут использоваться различные классификаторы, которые отличаются друг от друга охватом ПрО и степенью детализации при классификации этих областей, то есть можно сказать, что классификация ресурсов библиотеки основана на классификаторах и тезаурусе ПрО. При этом тезаурус может расширяться и пополняться новыми понятиями так же, как и классификатор.

Сказанное выводит на передний план ряд проблем, связанных с реализацией семантической библиотеки, в частности:

- как изменить модель данных и отразить эти изменения в системе;

- как влияют эти изменения на представление данных в интерфейсах пользователей и других потребителей информации (имеются в виду программные агенты, которые могут автоматически извлекать данные контента в машиночитаемом формате).

В целом эти проблемы сводятся к двум вариантам моделей данных и, соответственно, подходам их реализации.

Подход к разработке информационной системы семантической библиотеки *на предварительно жестко заданной модели данных* решает эти вопросы на этапе программной реализации, но любое изменение модели в таком случае требует дополнительной работы программистов. У подхода с жестко заданной моделью данных есть очевидные преимущества. В таких системах легче реализовать сложные взаимосвязи между ресурсами и проще построить «красивый» интерфейс для пользователя.

Подход, при котором *семантическая библиотека позволяет настраивать модель своего контента*, проще с точки зрения конечного пользователя, так как он получает возможность работы с моделью данных, не погружаясь в технические детали реализации. Это также ограничивает возможности моделирования «упрощая то, что можно упростить» в угоду возможности быстрой динамической настройки пользовательских интерфейсов под эти изменения.

В контексте этих проблем и на примере семантической библиотеки LibMeta

рассмотрим: (1) что такое модель данных семантической библиотеки; (2) как настраивается модель данных; (3) как связаны изменения в модели данных с интерфейсами пользователей.

3. ОСОБЕННОСТИ АРХИТЕКТУРЫ МОДЕЛИ ДАННЫХ БИБЛИОТЕКИ

Для построения семантической библиотеки LibMeta используются семантические технологии из стека Semantic Web[5]. Особое значение имеют онтологии[6], которые позволяют составить модель данных на основе ее понятий и отношений между ними. Понятия онтологии семантической библиотеки можно условно разделить на 2 группы:

- понятия (первого уровня), которые дают высокоуровневый взгляд на структуру контента библиотеки (например: ресурс, атрибут, тезаурус);

- понятия (второго уровня) конкретной ПрО (задачи математической физики, автор публикаций, тезаурус Обыкновенных Дифференциальных Уравнений (ОДУ), …).

Сами данные (например, задача Коши, Иванов И.И., …) представляются на основе заданных понятий.

Понятия первого уровня предоставляют возможность проектировать и реализовывать программный интерфейс семантической библиотеки, который, в свою очередь, позволяет описывать *понятия второго уровня*. Например, для того чтобы семантическая библиотека соответствовала ПрО «Математика», вводятся такие понятия, как «Публикация», «Персона», «Формула», которые связываются с тезаурусом ОДУ и «Математической энциклопедией» [14]. Математическая энциклопедия также является *экземпляром понятия «Тезаурус первого уровня»,* то есть в основе пользовательских интерфейсов настройки ПрО лежат *понятия первого уровня*.

На рис. 1 представлен в общем виде процесс формирования интерфейсов библиотеки на основе модели данных. Для этого процесса необходимо выполнить следующую последовательность шагов:

- Определить множество ресурсов *(Публикация, Автор, …);*

[5] https://www.w3.org/standards/semanticweb

[6] https://www.w3.org/standards/semanticweb/ontology

- Определить атрибуты *(название публикации, код (Номер) MSC, является автором, …);*
- Сформировать множество атрибутов для каждого ресурса;
- Сформировать интерфейсы системы на основе видов атрибутов:

– идентификационные атрибуты используются при выводе краткой информации в различных формах и при поиске дубликатов;

– поисковые атрибуты используются при формировании форм атрибутного поиска;

– описательные атрибуты используются для формирования форм редактирования и просмотра.

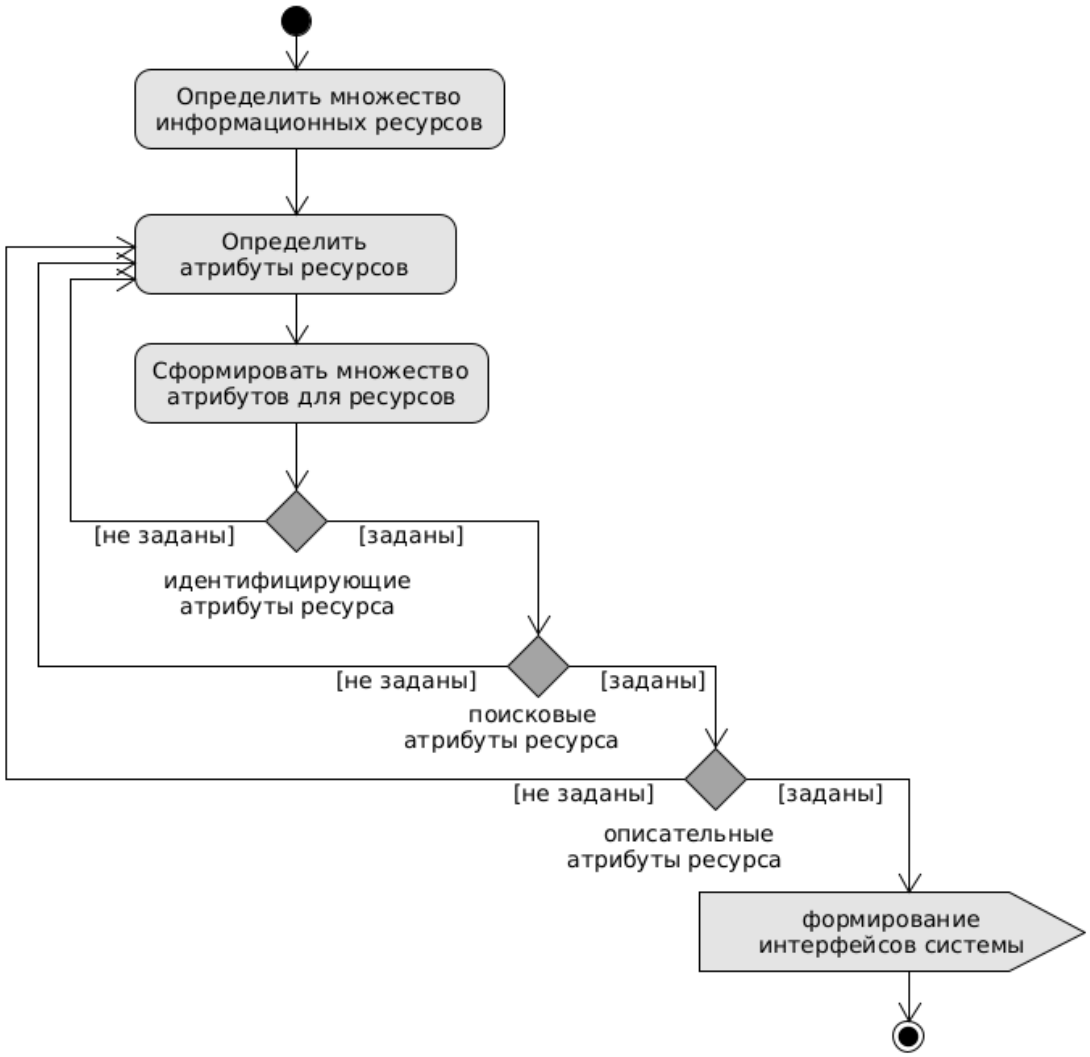


Рис. 1. Настройка интерфейсов

4. МОДЕЛЬ ДАННЫХ И ИНТЕРФЕЙСЫ

4.1. Особенности представления предметной области в библиотеке

В основе подхода настраиваемых интерфейсов находится набор понятий онтологии семантической библиотеки, необходимый для определения любой предметной области (ПрО) в рамках библиотеки. На основе этих понятий определяются специфические понятия предметной ПрО и связи между ними (рис. 2).

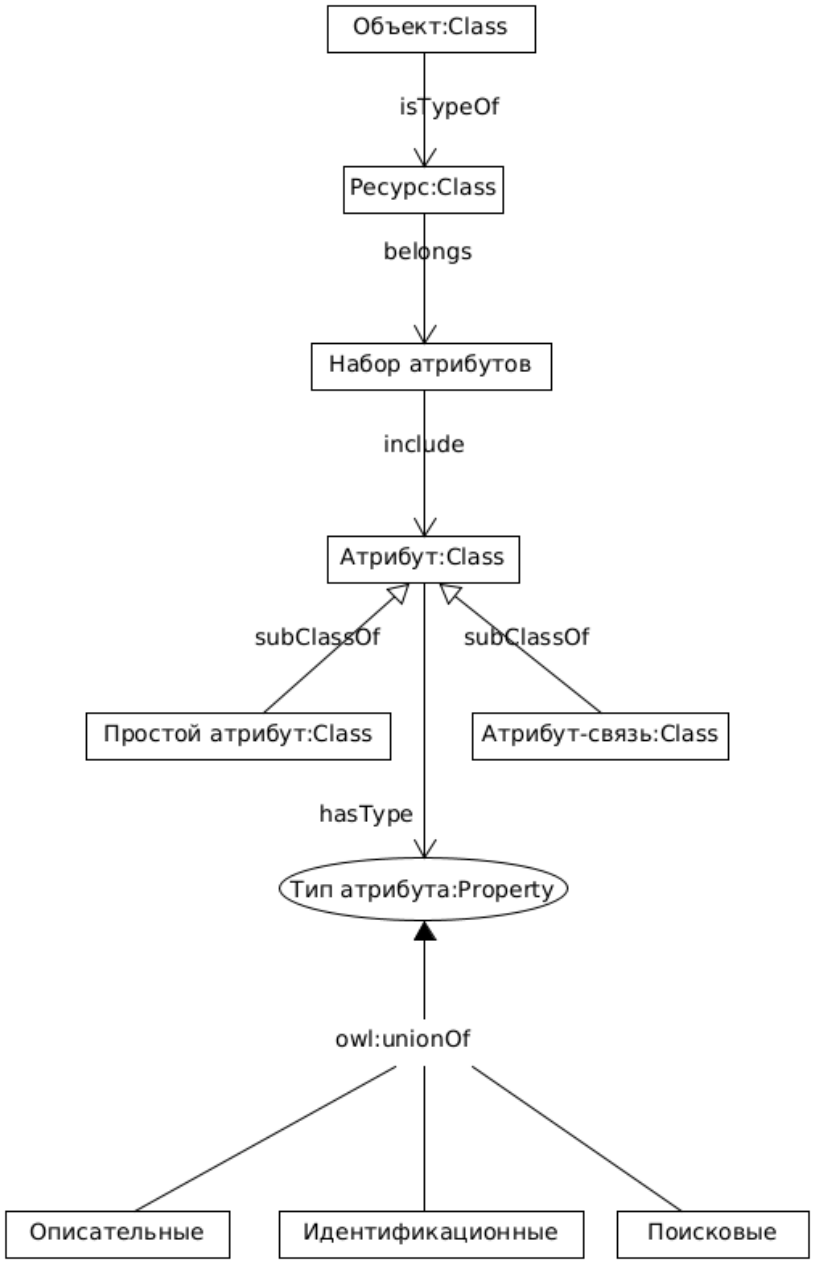


Рис. 2. Фрагмент онтологии

Рассмотрим пример, когда в качестве предметной области ПрО рассматривается подобласть математики «Обыкновенные дифференциальные уравнения». Ресурсами системы являются тезаурус обыкновенных дифференциальных уравнений и соответствующая ему литература. Для определения предметной области ПрО вводятся классы «Литература» и «Тезаурус ОДУ», являющиеся, с другой стороны, экземплярами класса «*Ресурс*» из онтологии, лежащей в основе библиотеки (рис. 2). Создание экземпляра ресурса происходит средствами системы (рис. 3).

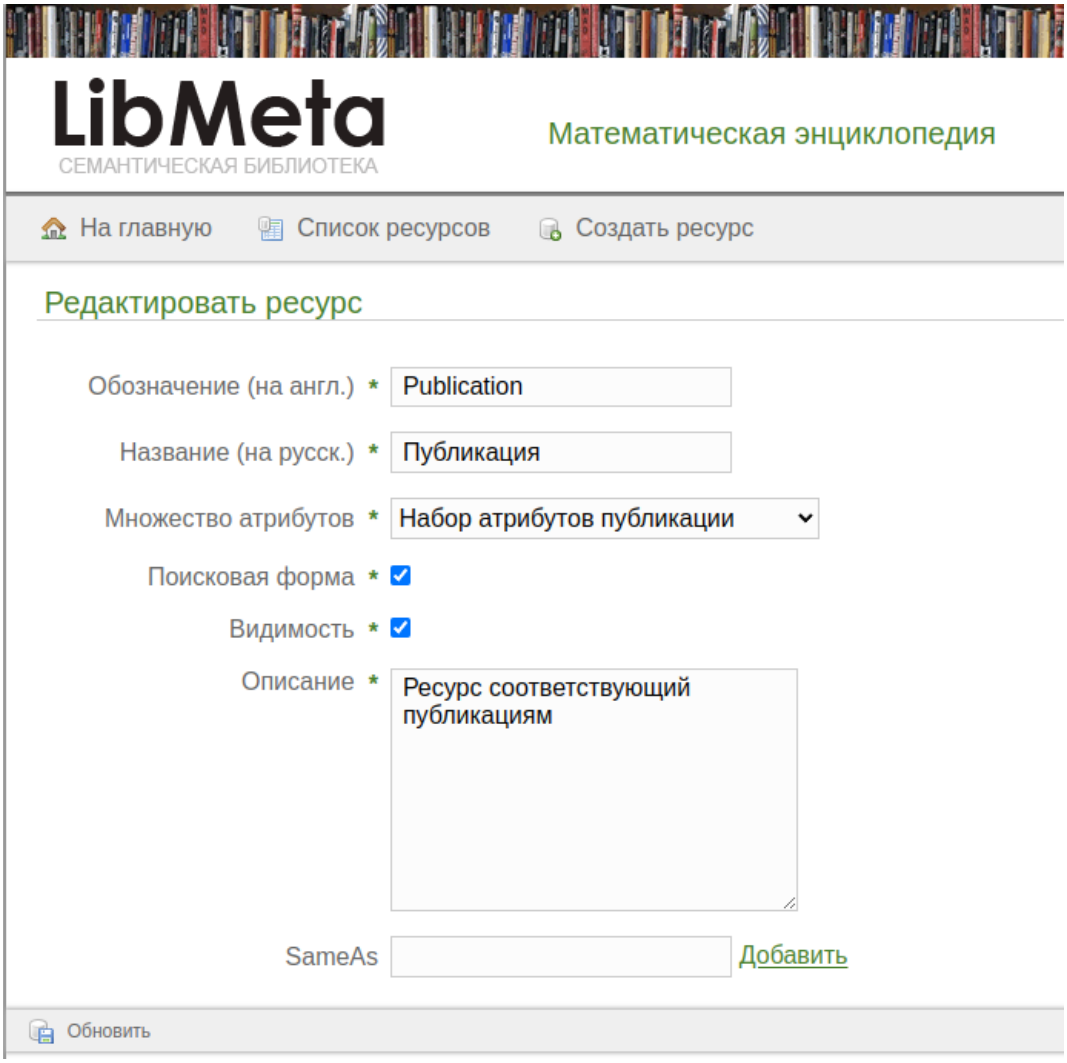


Рис. 3. Форма создания/редактирования ресурса

Каждый ресурс должен быть снабжен набором атрибутов, которые задают его структуру и в свою очередь делятся на простые атрибуты (такие, как строка,

число) и атрибуты связи, которые позволяют устанавливать связи с другими объектами в рамках библиотеки. Для этого создаются класс набора атрибутов и сами атрибуты на основе соответствующих понятий, представленных на рис. 1. В нашем примере для тезауруса ОДУ в набор его атрибутов добавляются такие атрибуты, как «математическая запись», «примечание», «литература». Наборы атрибутов различаются в зависимости от предметной области ПрО, а также могут изменяться в процессе развития библиотеки. На рис. 4 представлено понятие тезауруса с соответствующим набором значений созданных атрибутов.

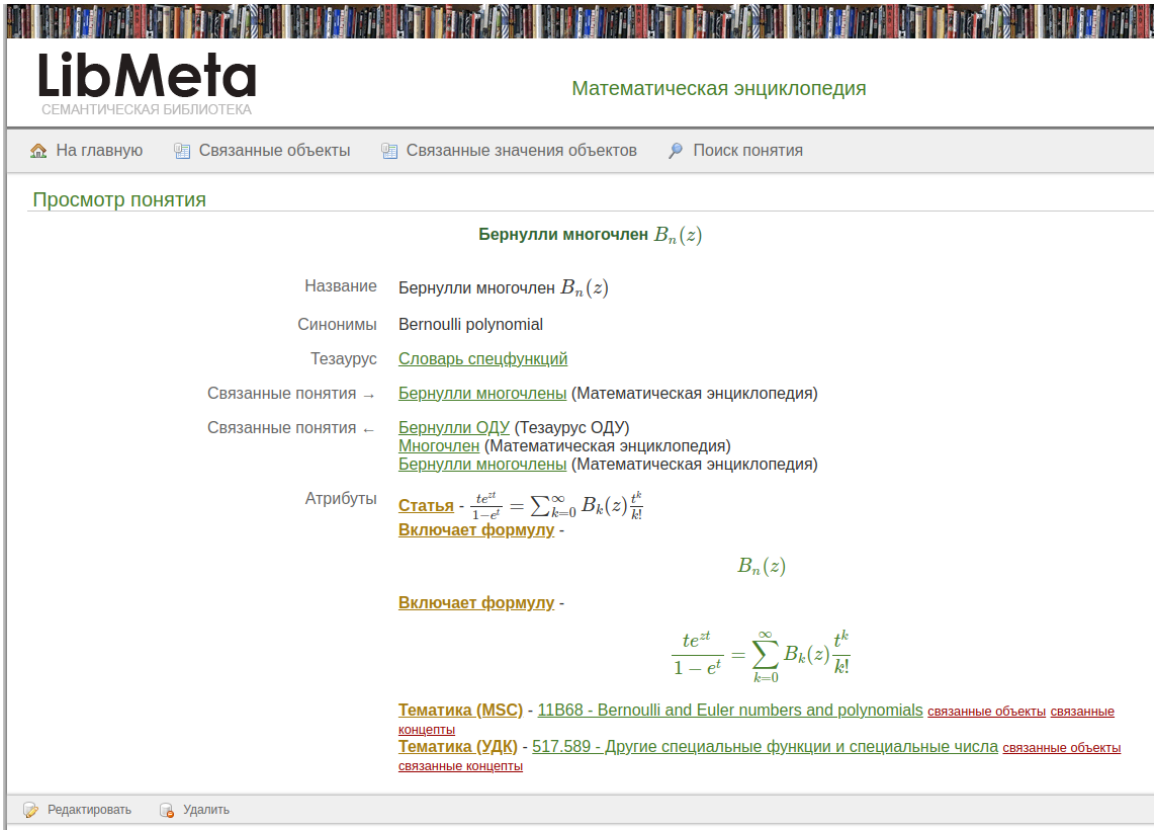


Рис. 4. Понятие тезауруса. Контент библиотеки

4.2. Настройка интерфейса

Рассмотрим пример настройки интерфейсов для атрибута-связи при определении атрибута «Номер MSC», который связывает литературу с классификатором MSC, при этом соответствующий ресурс *Литература* уже предварительно был создан. На основе онтологии в форме описания атрибута при указании атрибута «Номер MSC» и его типа «Таксономия» (рис. 5), использовав конкретный

классификатор MSC на форме редактирования публикации (рис. 6), получим элемент, который позволяет найти и «привязать» элемент классификатора к конкретной публикации, а на форме просмотра этой публикации (рис. 7) появится возможность перейти по ссылке и посмотреть связанные с ним объекты.

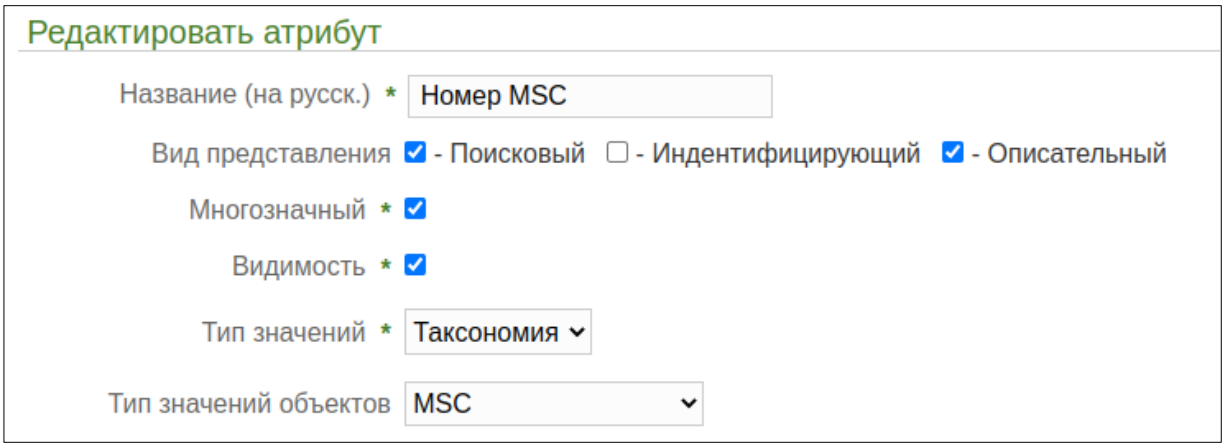


Рис. 5. Определение атрибута-связи

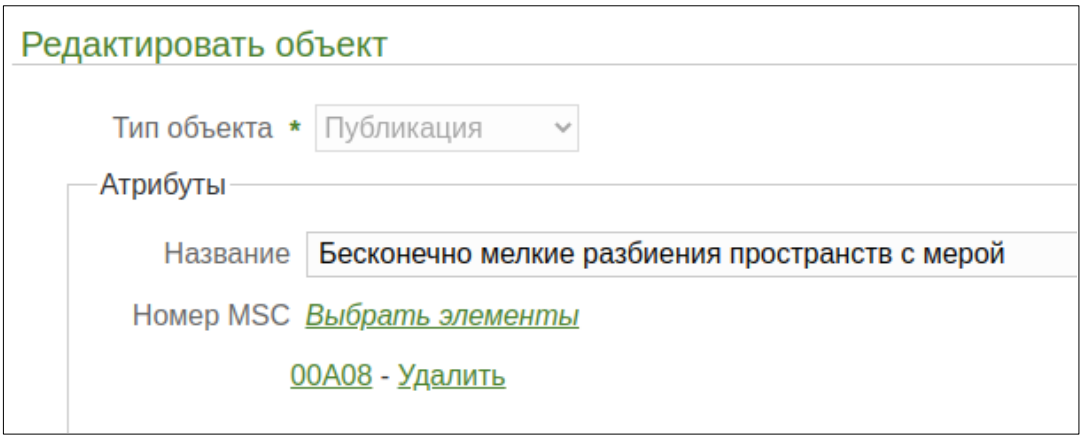


Рис. 6. Форма редактирования публикации

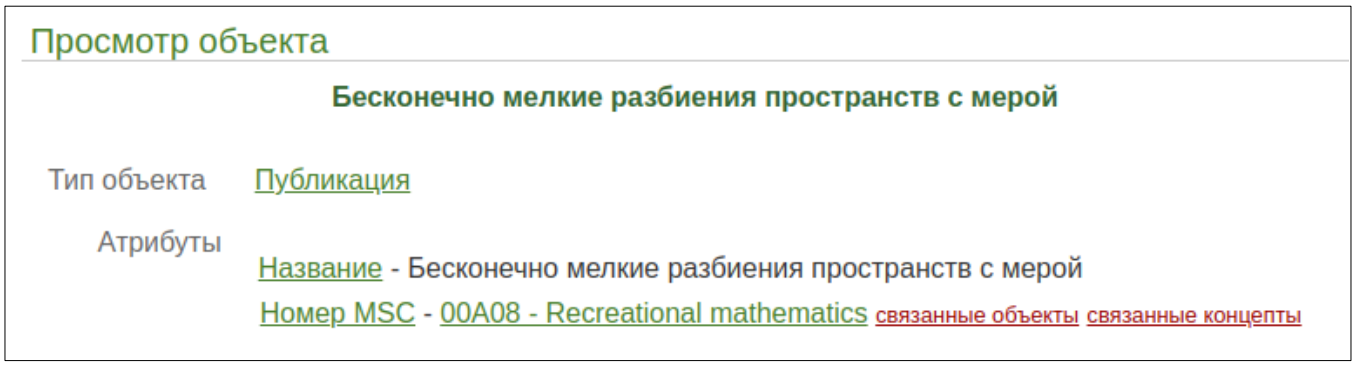


Рис. 7. Форма просмотра публикации

Аналогично можно настроить атрибут-связь и для тезауруса. Это позволит связать понятие тезауруса с любым объектом в контенте библиотеки.

Виды атрибута *поисковый, описательный, идентифицирующий* при определении атрибута (как простого, так и связи) в свою очередь позволят указать, *в каком виде представления информации* о ресурсе участвует тот или иной атрибут. Так как атрибут *«Номер MSC»* отмечен как *поисковый*, то соответствующий элемент появляется на *поисковой форме* (рис. 8).

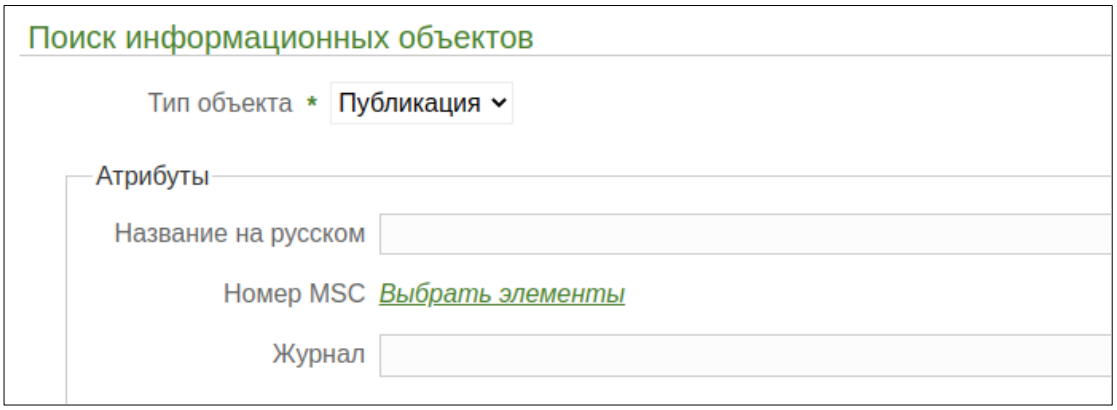


Рис. 8. Форма поиска публикации

5. ЗАКЛЮЧЕНИЕ

В модели семантической библиотеки введены понятия для описания содержимого библиотеки некоторой ПрО. Эти понятия позволяют сконструировать описание любых типов информационных ресурсов для выбранной области в рамках контента библиотеки. Информационные объекты, являющиеся непосредственно содержимым библиотеки, имеют распределенную природу, а именно, данные могут поступать из различных источников и агрегировать информацию из различных источников, что приводит к изменениям в существующей модели и их соответствующем отображении в интерфейсах библиотеки. Для реализации этого алгоритма как нельзя лучше подходит построение *адаптивных* интерфейсов системы на основе *модели данных* описания научных ресурсов. Это позволяет не ограничиваться при разработке строго фиксированным набором ресурсов. На примере семантической библиотеки LibMeta показан процесс формирования адаптивного интерфейса, соответствующего ПрО пользователя. Применение адаптивной модели позволяет понизить сложность (размерность) как самой модели данных, так и разрабатываемых на ее основе систем, ускоряя внедрение и развитие семантической библиотеки в практику исследований в конкретных ПрО.

Работа представлена в рамках выполнения темы госзадания «Математические методы анализа данных и прогнозирования» ФИЦ ИУ РАН.

СПИСОК ЛИТЕРАТУРЫ

1. *Sabatucci L., Seidita V., Cossentino M.* The Four Types of Self-adaptive Systems: A Metamodel. In: De Pietro G., Gallo L., Howlett R., Jai, L. (Eds). Intelligent Interactive Multimedia Systems and Services 2017. KES-IIMSS-18 2018. Smart Innovation, Systems and Technologies, 2018. Vol. 76 P. 440–450. Springer, Cham. https://doi.org/10.1007/978-3-319-59480-4_44.
URL: https://www.researchgate.net/publication/318132984_The_Four_Types_of_Self-adaptive_Systems_A_Metamodel (доступно 20.01.2023)

2. *Ferreira H., Correia F., Aguiar A.* Design for an Adaptive Object-Model Framework. An Overview. Proceedings of the 4th Workshop on Models@run.time, held at the ACM/IEEE 12th International Conference on Model Driven Engineering Languages and Systems (MoDELS'09**)** Denver, USA, October 5th, 2009. // CEUR Workshop Proceedings, M. Jeusfeld c/o Redaktion Sun SITE, Informatik V, RWTH Aachen (Aachen, Germany). 2009. Vol. 509. P. 71–80.
URL: http://ceur-ws.org/Vol-509/paper_13.pdf (доступно 20.01.2023)

3. *Yoder J.W., Johnson R*. The Adaptive Object-Model Architectural Style. 2002. URL: https://www.researchgate.net/publication/220864957.

4. *Fowler M., Lewis J.* Microservices. 2014. [Online].
URL: http://martinfowler.com/articles/microservices.html (доступно 20.01.2023)

5. *Andrade B., Santos S., Silva A. R.* From Monolith to Microservices: Static and Dynamic Analysis Comparison. 2022. URL: https://10.48550/arXiv.2204.11844 Corpus ID: 248392322 (доступно 20.01.2023)

6. *Santos N., Silva A.R.* A Complexity Metric for Microservices Architecture // Computer Science. IEEE International Conference on Software Architecture (ICSA). Salvador, Brazil, 2020. P. 169–178, https://doi.org/10.1109/ICSA47634.2020.00024.

7. *Mascardi V., Cordi V., Rosso P.* A Comparison of Upper Ontologies. Conference: WOA 2007: Dagli Oggetti agli Agenti. 8th AI*IA/TABOO Joint Workshop "From Objects to Agents": Agents and Industry: Technological Applications of Software Agents, 24–25 September 2007, Genova, Italy. 2007.

8. *Katsis Y., Papakonstantinou Y.* View-based data integration // Encyclopedia of Database Systems. 2009. P. 3332–3339.

9. *Xu L., Embley D.W.* Combining the Best of Global-as-View and Local-as-View for Data Integration //ISTA. 2004. Vol. 48. P. 123–136.

10. *Noy N.*F. Semantic integration: a survey of ontology-based approaches // ACM Sigmod Record. 2004. Vol. 33. No. 4. P. 65–70.

11. *Zhao L., Ichise R.* Ontology integration for linked data // Journal on Data Semantics. 2014. Vol. 3. No. 4. P. 237–254.

12. *Serebryakov V.A., Ataeva O.*M. Ontology Based Approach to Modeling of the Subject Domain ‘‘Mathematics’’ in the Digital Library // Lobachevskii Journal of Mathematics. 2021. V. 42. No. 8. P. 1920–1934.

13. *Ataeva O.M., Serebryakov V.A., Tuchkova N.P.* Creation of query expansion based on the subject domain thesaurus in the ontology of knowledge of the semantic library // Scientific service on the Internet: Proceedings of the XXI All-Russian Scientific Conference (September 23–28, 2019, Novorossiysk). P. 63–74. https://doi.org/10.20948/abrau-2019-12.

14. *Ataeva O., Serebryakov V., Tuchkova N.* Creating the Applied Subject Area Ontology by Means of the Content of the Digital Semantic Library // Lobachevskii Journal of Mathematics, 2022. V. 43. No. 7. P. 1795–1804. https://doi.org/10.1134/S1995080222100043

MODELING AN ADAPTIVE INTERFACE USING SEMANTIC ONTOLOGY RELATIONS

O. M. Ataeva[1] [0000-0003-0367-5575], V. A. Serebriakov[2] [0000-0003-1423-621X],

N. P. Tuchkova[3] [0000-0001-5357-9640]

[1,2,3]*Dorodnicyn Computing Centre FRC CSC RAS, Moscow*

[1]oli@ultimeta.ru, [2]serebr@ultimeta.ru, [3]natalia_tuchkova@mail.ru

Abstract

The work is devoted to the problem of customizing the user interfaces of an information system that integrates data. An adaptive interface serves as one of the means of organizing the presentation of subject domain data. The issue of using the

semantic relations of ontology to select data corresponding to the objectives of the study is investigated. A model of an adaptive interface is considered, which allows the most accurate reflection of the needs of a researcher within a particular subject domain. It is shown how the adaptive interface is formed by means of the semantic library model.

Keywords: *ontology, adaptive interface, subject domain, data model*

REFERENCES

1. *Sabatucci L., Seidita V., Cossentino M.* The Four Types of Self-adaptive Systems: A Metamodel. In: De Pietro G., Gallo L., Howlett R., Jai, L. (Eds). Intelligent Interactive Multimedia Systems and Services 2017. KES-IIMSS-18 2018. Smart Innovation, Systems and Technologies, 2018. Vol. 76 P. 440–450. Springer, Cham. https://doi.org/10.1007/978-3-319-59480-4_44.
URL: https://www.researchgate.net/publication/318132984_The_Four_Types_of_Self-adaptive_Systems_A_Metamodel (доступно 20.01.2023)

2. *Ferreira H., Correia F., Aguiar A.* Design for an Adaptive Object-Model Framework. An Overview. Proceedings of the 4th Workshop on Models@run.time, held at the ACM/IEEE 12th International Conference on Model Driven Engineering Languages and Systems (MoDELS'09**)** Denver, USA, October 5th, 2009. // CEUR Workshop Proceedings, M. Jeusfeld c/o Redaktion Sun SITE, Informatik V, RWTH Aachen (Aachen, Germany). 2009. Vol. 509. P. 71–80.
URL: http://ceur-ws.org/Vol-509/paper_13.pdf (доступно 20.01.2023)

3. *Yoder J.W., Johnson R*. The Adaptive Object-Model Architectural Style. 2002. URL: https://www.researchgate.net/publication/220864957.

4. *Fowler M., Lewis J.* Microservices. 2014. [Online].
URL: http://martinfowler.com/articles/microservices.html (доступно 20.01.2023)

5. *Andrade B., Santos S., Silva A. R.* From Monolith to Microservices: Static and Dynamic Analysis Comparison. 2022. URL: https://10.48550/arXiv.2204.11844 Corpus ID: 248392322 (доступно 20.01.2023)

6. *Santos N., Silva A.R.* A Complexity Metric for Microservices Architecture // Computer Science. IEEE International Conference on Software Architecture (ICSA). Salvador, Brazil, 2020. P. 169–178, https://doi.org/10.1109/ICSA47634.2020.00024.

7. *Mascardi V., Cordi V., Rosso P.* A Comparison of Upper Ontologies. Conference: WOA 2007: Dagli Oggetti agli Agenti. 8th AI*IA/TABOO Joint Workshop "From Objects to Agents": Agents and Industry: Technological Applications of Software Agents, 24–25 September 2007, Genova, Italy. 2007.

8. *Katsis Y., Papakonstantinou Y.* View-based data integration // Encyclopedia of Database Systems. 2009. P. 3332–3339.

9. *Xu L., Embley D.W.* Combining the Best of Global-as-View and Local-as-View for Data Integration //ISTA. 2004. Vol. 48. P. 123–136.

10. *Noy N.*F. Semantic integration: a survey of ontology-based approaches // ACM Sigmod Record. 2004. Vol. 33. No. 4. P. 65–70.

11. *Zhao L., Ichise R.* Ontology integration for linked data // Journal on Data Semantics. 2014. Vol. 3. No. 4. P. 237–254.

12. *Serebryakov V.A., Ataeva O.*M. Ontology Based Approach to Modeling of the Subject Domain ‘‘Mathematics’’ in the Digital Library // Lobachevskii Journal of Mathematics. 2021. V. 42. No. 8. P. 1920–1934.

13. *Ataeva O.M., Serebryakov V.A., Tuchkova N.P.* Creation of query expansion based on the subject domain thesaurus in the ontology of knowledge of the semantic library // Scientific service on the Internet: Proceedings of the XXI All-Russian Scientific Conference (September 23–28, 2019, Novorossiysk). P. 63–74. https://doi.org/10.20948/abrau-2019-12.

14. *Ataeva O., Serebryakov V., Tuchkova N.* Creating the Applied Subject Area Ontology by Means of the Content of the Digital Semantic Library // Lobachevskii Journal of Mathematics, 2022. V. 43. No. 7. P. 1795–1804. https://doi.org/10.1134/S1995080222100043

СВЕДЕНИЯ ОБ АВТОРАХ

АТАЕВА Ольга Муратовна – научный сотрудник Вычислительного центра им. А.А. Дородницына ФИЦ ИУ РАН, кандидат техн. наук, специалист в области системного программирования и баз данных.

Olga Muratovna ATAEVA – researcher of the of Dorodnicyn computing center FRC SCS RAS, PhD, expert in the field of system programming and databases.

email: oli@ultimeta.ru

ORCID: 0000-0003-0367-5575

СЕРЕБРЯКОВ Владимир Алексеевич – специалист в области теории формальных языков и её приложений, доктор физ.-мат. наук, профессор, зав. отделом Вычислительного центра им. А.А. Дородницына ФИЦ ИУ РАН. Руководитель и участник разработки ряда известных программных проектов, в частности, ИСИР и ИСИР РАН, «Научный портал РАН».

Vladimir Alekseevich SEREBRIAKOV – expert in the field of theory of formal languages and its applications, doctor of sciences, professor, head of Dorodnicyn computing center FRC SCS RAS department. Head and participant in the development of a number of well-known program projects, in particular, ISIR and ISIR RAS, Scientific portal RAS.

email: serebr@ultimeta.ru

ORCID: 0000-0003-1423-621X

ТУЧКОВА Наталия Павловна – старший научный сотрудник Вычислительного центра им. А.А. Дородницына ФИЦ ИУ РАН, кандидат физ.-мат. наук, окончила ВМиК МГУ им. М.В. Ломоносова. Специалист в области алгоритмических языков и информационных технологий.

Natalia Pavlovna TUCHKOVA – senior researcher of Dorodnicyn computing center FRC SCS RAS, PhD in physics with a math degree, graduated from CS Faculty of Lomonosov MSU. The expert in the field of algorithmic languages and information technologies.

email: natalia_tuchkova@mail.ru

ORCID: 0000-0001-5357-9640

Материал поступил в редакцию 30 января 2023 года

УДК 519.713

КОНЕЧНО-АВТОМАТНЫЕ МЕТОДЫ СИНТЕЗА ТЕСТОВ С ГАРАНТИРОВАННОЙ ПОЛНОТОЙ ДЛЯ ВХОДО-ВЫХОДНЫХ ПОЛУАВТОМАТОВ

И. Б. Бурдонов[1] [0000-0001-9539-7853], Н. В. Евтушенко[2] [0000-0002-4006-1161], А. С. Косачев[3] [0000-0001-5316-3813]

[1, 2, 3]Институт системного программирования им. В.П. Иванникова Российской академии наук, г. Москва

[1]igor@ispras.ru, [2]evtushenko@ispras.ru, [3]kos@ispras.ru

Аннотация

Рассмотрена проблема использования конечно-автоматных методов для построения конечных тестов с гарантированной полнотой для входо-выходных полуавтоматов. Предложен способ построения конечного автомата, соответствующего полуавтомату-спецификации, и показано, что конечные тесты, построенные по такому автомату, подаваемые на вход-выходной полуавтомат при выполнении специальных таймаутов, являются полными относительно различных моделей неисправности.

***Ключевые слова**: входо-выходной полуавтомат, конечный автомат, модель неисправности, полный тест*

ВВЕДЕНИЕ

Построение тестов на основе формальных моделей (MBT в англоязычной литературе) [1–8] в настоящее время широко применяется для различных дискретных и гибридных систем, поскольку такие тесты позволяют гарантировать отсутствие критических ошибок в реализации системы. В этом случае обычно предполагается, что поведение спецификации системы и проверяемой реализации описано одной и той же формальной моделью, и определено правило, согласно которому проверяемая реализация *соответствует* спецификации, т. е. не содержит ошибок. Для модели конечного автомата, в которой после каждого входного воздействия ожидается выходной сигнал, разработано большое количество

методов синтеза полных конечных тестов относительно различных моделей неисправности (см., например, [1–4, 9–11]), т. е. тестов, обнаруживающих каждую некорректную реализацию из заданного конечного класса без явного перечисления возможных реализаций. Однако достаточно часто в качестве спецификации дискретных систем рассматривается модель входо-выходного полуавтомата, в которой после входного воздействия может отсутствовать выходная реакция или может появиться последовательность таких реакций. Возможно также наличие ненаблюдаемого действия, что может привести к возникновению тупиковых ситуаций. Несмотря на достаточно большое количество публикаций по синтезу проверяющих тестов на основе такой модели [5, 6], конечные тесты с гарантированной полнотой строятся по модели неисправности, в которой все полуавтоматы-реализации явно перечислены, или для наблюдения доступны состояния тестируемой системы, или бесконечный тест ограничивается некоторым случайным образом, в результате чего полнота тестирования остается неизвестной. В работе [12] нами предложены правила для подачи входных последовательностей на входо-выходной полуавтомат, которые позволяют избежать состязаний между входными и выходными воздействиями и тупиковых ситуаций. В этом случае по входо-выходному полуавтомату можно построить подходящий конечный автомат [13] и соответственно синтезировать тесты с гарантированной полнотой известными конечно автоматными методами. В настоящей работе мы подробно рассматриваем две модели неисправности на основе входо-выходных полуавтоматов, для которых возможен синтез таких конечных полных проверяющих тестов.

Структура статьи следующая. Второй раздел содержит необходимые определения и обозначения. В третьем разделе введены понятия модели неисправности, в то время как четвертый раздел посвящен методам синтеза полных проверяющих тестов относительно введенной модели неисправности. В заключении подведены итоги работы и обсуждены перспективы дальнейших научных исследований.

1. ОПРЕДЕЛЕНИЯ И ОБОЗНАЧЕНИЯ

Конечный входо-выходной *полуавтомат* (или далее просто *полуавтомат*) [5, 6] есть четверка $\boldsymbol{S}=(S,s_0,I,O,h_S)$, где S – конечное непустое множество со-

стояний с выделенным начальным состоянием s_0, I – конечное непустое множество входных действий, O – конечное непустое множество выходных действий, $I \cap O = \varnothing$, и $h_S \subseteq S \times (I \cup O) \times S$ есть отношение переходов. В полуавтомате есть переход из состояния s в состояние s' под действием символа a, если и только если тройка $(s,a,s') \in h_S$. Полуавтомат *наблюдаемый*, если в каждом состоянии для каждого действия определено не более одного перехода [12]. Полуавтомат является *недетерминированным* (по выходным символам), если в некотором состоянии определены переходы по нескольким выходным действиям [12]; в противном случае входо-выходной полуавтомат *детерминированный*. В настоящей работе мы рассматриваем только наблюдаемые полуавтоматы. Входной символ из I *определен* в состоянии s, если в этом состоянии есть переход под действием этого входного символа. Полуавтомат называется *полностью определенным* (по входным символам), если в каждом состоянии определен переход по любому входному действию; иначе полуавтомат называется *частично определенным*. Полуавтомат является трассовой моделью и описывает поведение моделируемой системы на трассах (последовательностях действий) из алфавита $I \cup O$. В состоянии s последовательность из $I \cup O$ является *допустимой*, если ее можно получить посредством последовательных переходов из этого состояния. Для *эквивалентных* полностью определенных полуавтоматов S и P (обозначение $P \cong S$) множества их трасс совпадают. Если P есть *редукция* S (обозначение $P \leq S$), то множество трасс полуавтомата P есть подмножество трасс полуавтомата S. Полуавтомат может иметь состояния, в которых нет переходов, помеченных выходными действиями; такие состояния часто называют *устойчивыми,* поскольку полуавтомат может оставаться в таком состоянии неограниченно долгое время, пока не будет подан входной сигнал. К таким состояниям, в частности, относятся тупиковые состояния, т. е. состояния, в которых не определено ни одного перехода. Трасса в состоянии s называется *полной*, если финальное состояние трассы является устойчивым. По определению, после выполнения полной трассы полуавтомат может оставаться в достигнутом состоянии до подачи нового входного воздействия. Для внешнего наблюдения перехода полуавтомата в устойчивое состояние вводится специальный «молчащий» выходной символ $\delta \notin I \cup O$ (англ. quiescence) [5]. Таким образом, можно полагать, что в каждом устойчивом состоянии полуавтомата есть петля, помеченная

символом δ, который рассматривается как выходной символ, и расширенный полуавтомат с выходным алфавитом $O \cup \{\delta\}$ обозначается S^{δ}. Соответственно, трасса σ полуавтомата S в состоянии *s* является полной, если и только если в полуавтомате S^{δ} в состоянии *s* есть трасса σδ, так называемая δ-трасса. Фактически мы этим подчеркиваем, что ни один выходной символ из *O* не может появиться после трассы σ. По определению, из трассы полуавтомата S^{δ} можно получить трассу S после удаления δ, и обратно, после добавления любого количества символов δ после любого полного префикса трассы σ полуавтомата S получается трасса полуавтомата S^{δ}.

В *конечном автомате* $M=(M,m_0,I,O,h_M)$ [14] переходы в каждом состоянии помечены парами *i*/*o*, соответственно отношение переходов содержит четверки вида (*m*,*i*,*o*,*m*'). Автомат называется *детерминированным*, если в любом состоянии определено не более одного перехода по каждому входному действию; в противном случае автомат *недетерминированный*. Автомат называется *наблюдаемым*, если в каждом состоянии для каждой входо-выходной пары действия определено не более одного перехода; в противном случае автомат *ненаблюдаемый*. Автомат называется *полностью определенным*, если в каждом состоянии определен переход по любому входному символу; иначе полуавтомат называется *частичным* или *частично определенным*. Чтобы вычислить возможную выходную реакцию автомата в состоянии *s* на входную последовательность $i_1 \ldots i_k$, достаточно из состояния *s* пройти последовательно по переходам, которые помечены входо-выходной парой с соответствующим входным символом.

2. ТЕСТИРОВАНИЕ НА ОСНОВЕ ВХОДО-ВЫХОДНЫХ ПОЛУАВТОМАТОВ

Процесс тестирования интерактивных систем на основе формальных моделей (англ. Model Based Testing, MBT) обычно содержит три этапа: 1) на тестируемую реализацию подается тестовая (-ые) последовательность (-ти); 2) наблюдается выполняемая трасса; и 3) принимается решение о соответствии тестируемой реализации заданной спецификации. Процесс тестирования называется *безусловным* (англ. *preset*), если множество входных последовательностей определено заранее и не изменяется в процессе тестирования. Тестирование называется *адаптивным*, если следующий входной символ в тестовой последовательности

зависит от реакции тестируемой реализации на предыдущие входные воздействия. Подобно классическим конечным автоматам [15], для входо-выходных полуавтоматов можно ввести модель неисправности, которая также является тройкой FM=<S,⊳,Ω>. В этой тройке S – спецификация системы в виде входо-выходного полуавтомата, Ω – множество входо-выходных полуавтоматов, описывающих поведение любой предъявленной для тестирования реализации, входной и выходной алфавиты которой совпадают с таковыми для спецификации S, и ⊳ – отношение конформности между полуавтоматами S и P∈Ω, определяющее «правильность» тестируемой реализации относительно спецификации, обозначение S⊳P. Множество трасс, допустимых в спецификации, является *полным* тестом относительно модели неисправности FM, если и только если при подаче теста на входо-выходной полуавтомат P из Ω вердикт ‘pass’ выдается для полуавтоматов, конформных спецификации, и только для них.

Для входо-выходных полуавтоматов методы синтеза полных тестов разработаны для различных отношений конформности [5, 6], но построенный полный тест является конечным только для случая, когда множество тестируемых реализаций задано явным перечислением (модель «белого ящика») или при возможности наблюдения состояний тестируемой системы, т. е. для очень узкого класса моделей неисправности.

В данной работе, подобно [12, 13], мы вводим специальные ограничения на работу системы, поведение которой описано входо-выходным полуавтоматом, и покажем, что при выполнении этих ограничений можно перейти к тестам, которые суть последовательности входных символов и таймаутов, используемых для ожидания выходного символа. В этом случае по модели неисправности для входо-выходных полуавтоматов строится модель неисправности для классических автоматов, и полный тест, построенный относительно этой модели, является полным относительно исходной модели неисправности для входо-выходных полуавтоматов. В следующем разделе мы проиллюстрируем наш подход для двух наиболее часто используемых отношений конформности для входо-выходных полуавтоматов.

($T_{вх}$,$T_{вых}$)-входо-выходные полуавтоматы. Мы далее предположим, что поведение системы, описанной входо-выходным полуавтоматом, удовлетворяет

следующим правилам. После достижения системой текущего состояния она ожидает входной символ в пределах таймаута $T_{вх}$. Если входной символ подан, то система переходит в следующее предписанное состояние, таймер «сбрасывается», и система ожидает следующий входной символ. Если входной символ не появился в пределах таймаута $T_{вх}$, то он «сбрасывается», система начинает подготовку допустимого в состоянии выходного символа, который должен появиться в пределах таймаута $T_{вых}$, и никакие входные символы в этот период на систему не подаются. Если выходной сигнал появляется, то таймер «сбрасывается», и система готова к принятию следующего входного сигнала. Если выходной символ не появляется в течение $T_{вых}$, то полагаем, что полуавтомат выдает «молчащий» символ δ, и таймер «сбрасывается». Полуавтоматы, работающие по таким правилам, будем называть входо-выходными ($T_{вх}$,$T_{вых}$)-*полуавтоматами*. Рассмотрим подачу входной последовательности ?*i*?*i* на входо-выходной полуавтомат на рис. 1а. Входной символ ?*i* подается в пределах таймаута $T_{вх}$, таймер «сбрасывается», и на систему подается второй входной символ ?*i*. После этого входной символ в пределах таймаута $T_{вх}$ не подается, таймер «сбрасывается», и система начинает подготовку допустимого в состоянии 2 выходного символа $!o_2$, который должен появиться в пределах таймаута $T_{вых}$.

Для преобразования входо-выходного ($T_{вх}$,$T_{вых}$)-полуавтомата S^{δ} в конечный автомат $M^{\delta\omega}{}_S$ введем специальный входной символ $\omega \notin I$, соответствующий отсутствию входного символа, т. е. ожиданию выходного символа. Соответственно, по такому полуавтомату можно построить конечный автомат с тем же множеством трасс [13], по которому можно построить полные тесты относительно различных моделей неисправности известными конечно автоматными методами. Множество состояний конечного автомата $M^{\delta\omega}{}_S$ совпадает с таковым для полуавтомата S. Для полуавтомата S с входным и выходным алфавитами *I* и *O* входной и выходной алфавиты автомата суть ($I\cup\{\omega\}$) и ($O\cup\{\delta\}$). Автомат $M^{\delta\omega}{}_S$ строится по следующим правилам:

– существует переход из состояния *s* в состояние *q* с выходным символом δ под действием входного символа $i \in I$, если и только если в полуавтомате S существует переход из *s* в *q* под действием *i*;

– существует переход из состояния *s* в состояние *q* с выходным символом *o*

под действием входного символа ω, если и только если в полуавтомате S существует переход из *s* в *q* с выходным символом *o*;

– в состоянии *s* есть петля, помеченная парой ω/δ, если и только если *s* есть устойчивое состояние полуавтомата S.

В качестве примера рассмотрим полуавтомат $\boldsymbol{S}$ на рис. 1, в котором входные символы помечены знаком ?, а выходные символы – знаком !, и соответствующий ему автомат $M^{\delta\omega}{}_S$.

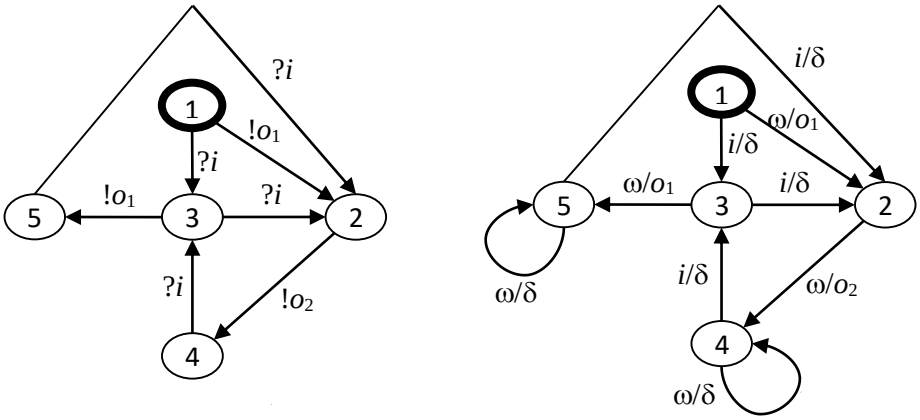


Рис. 1. Полуавтомат $\boldsymbol{S}$ и соответствующий ему конечный автомат $M^{\delta\omega}{}_S$.

По построению для автомата $M^{\delta\omega}{}_S$ имеют место следующие утверждения.

Утверждение 1. Полуавтомат S^{δ} содержит трассу σ, если и только если автомат $M^{\delta\omega}{}_S$ содержит трассу, построенную по σ следующим образом: сначала перед каждым выходным символом $o\in(O\cup\{\delta\})$ вставлен символ ω; после этого после каждого входного символа из алфавита *I* добавлен символ δ.

Утверждение 2. Если S – полностью определенный по входным символам входо-выходной ($T_{вх}$,$T_{вых}$)-полуавтомат, то полуавтомат S^{δ} имеет выходную реакцию $\beta_1\beta_2\ldots\beta_k\beta_{k+1}$, $\beta_j\in(O\cup\{\delta\})^*$, $|\beta_j|=t_j$, на входную последовательность $\alpha=\omega^{t1}i_1\omega^{t2}\ldots\omega^{t(k)}i_k\omega^{t(k+1)}$, $t(j)\in\{0, 1, \ldots\}$, j = 1, …, k+1, если и только в автомате $M^{\delta\omega}{}_S$ существует выходная реакция $\beta_1\delta\beta_2\ldots\beta_k\delta\beta_{k+1}$ на входную последовательность α.

Утверждение достаточно просто доказывается по индукции по числу *k*, поскольку выходная реакция в полуавтомате S^{δ} существует только в случае, когда входной символ не подается в ограниченный промежуток времени, т. е. подается

«искусственно введенный» входной символ ω. Если полуавтомат S^{δ} имеет выходную реакцию $\beta_1\beta_2 \in (O \cup \{\delta\})^*$, $|\beta_1|=t_1$, $|\beta_2|=t_2$, на входную последовательность $\alpha=\omega^{t1} i_1 \omega^{t2}$, то, по определению, автомат $M^{\delta\omega}{}_S$ на данную входную последовательность имеет выходную реакцию $\beta_1\delta\beta_2$.

В качестве примера рассмотрим полуавтомат на рис. 1а. Пусть на вход полуавтомата подается входная последовательность ?*i*?*i*, в которой оба входных символа подаются в пределах таймаута $T_{вх}$. После подачи второго входного символа ?*i* таймер «сбрасывается», и система начинает подготовку допустимого в состоянии 2 выходного символа $!o_2$, который должен появиться в пределах таймаута $T_{вых}$, т. е. полуавтомат выдает $!o_2$ при подаче входной последовательности ?*i*?*i*?ω. Непосредственно проверкой можно убедиться, что автомат $M^{\delta\omega}{}_S$ на рис. 1б имеет выходную последовательность $\delta\delta o_2$ на входную последовательность *i*?*i*?ω, которая превращается в последовательность o_2 после удаления «искусственных» выходных символов δ на входной символ *i*.

Таким образом, полуавтомат S^{δ} можно преобразовать в конечный автомат, наблюдаемые трассы которого будут совпадать с таковыми для исходного входо-выходного ($T_{вх}$,$T_{вых}$)-полуавтомата S при выполнении ограничений для таймаутов при подаче входных и выдаче выходных последовательностей. Соответственно, для таких полуавтоматов в ряде случаев понятие модели неисправности и проверяющего теста можно преобразовать таким образом, чтобы воспользоваться известными методами построения тестов с гарантированной полнотой для конечных автоматов. В следующем разделе более подробно рассмотрены две такие модели неисправности.

3. ПОСТРОЕНИЕ ПОЛНЫХ ПРОВЕРЯЮЩИХ ТЕСТОВ ДЛЯ ВХОДО-ВЫХОДНЫХ ПОЛУАВТОМАТОВ

Модель неисправности, в которой полуавтомат-спецификация детерминированный по выходам

Рассмотрим модель неисправности FM=<S,≅,Ω>, в которой спецификация системы S и все элементы множества Ω суть полностью определенные по входным символам ($T_{вх}$,$T_{вых}$)-полуавтоматы для выбранных значений $T_{вх}$ и $T_{вых}$, и ≅ есть отношение (трассовой) эквивалентности между полуавтоматами S^{δ} и P^{δ}. В этом

случае *проверяющим тестом* относительно FM называется последовательность входных символов и символа ω, который означает ожидание выходного символа или «молчащего» символа δ в пределах таймаута $T_{вых}$. Тест подается на тестируемый полуавтомат следующим образом.

Входной символ из алфавита I подается достаточно «быстро» в пределах таймаута $T_{вх}$, и таймер «сбрасывается». Если в тесте встречается символ ω, то никакой входной символ на тестируемую систему не подается, и в течение промежутка времени $T_{вых}$ начинает формироваться и ожидается выходной символ или «молчащий» символ δ. Тест называется *полным* относительно $<S,\cong,\Omega>$, если для любого полуавтомата $P\in\Omega$, такого что S^{δ} и P^{δ} не являются эквивалентными, в тесте есть последовательность, множества реакций (реакции) на которую полуавтоматов S^{δ} и P^{δ} различны.

Рассмотрим автоматную модель $FM_{FSM}=<M^{\delta\omega}{}_S,\cong_{FSM},\Omega^{\delta\omega}>$, в которой спецификация системы есть конечный автомат $M^{\delta\omega}{}_S$, множество $\Omega^{\delta\omega}$ содержит все автоматы P^{δ}, $P\in\Omega$, и $\cong$ есть отношение (трассовой) эквивалентности между автоматами $M^{\delta\omega}{}_S$ и $M^{\delta\omega}{}_P$.

Утверждение 3. Проверяющий тест является полным относительно модели неисправности $FM=<S,\cong,\Omega>$, в которой спецификация системы S и все элементы множества Ω суть полностью определенные по входным символам ($T_{вх}$,$T_{вых}$)-полуавтоматы для выбранных значений $T_{вх}$ и $T_{вых}$, и $\cong$ есть отношение (трассовой) эквивалентности между полуавтоматами S^{δ} и P^{δ}, если и только если тест является полным относительно автоматной модели неисправности FM_{FSM} $<M^{\delta\omega}{}_S,\cong_{FSM},\Omega^{\delta\omega}>$.

Проверяющие тесты относительно моделей неисправности $<S,\cong,\Omega>$ и $<M^{\delta\omega}{}_S,\cong_{FSM},\Omega^{\delta\omega}>$ содержат входные последовательности в алфавите $I\cup\{\omega\}$. Проверяющий тест *TS* полный относительно $FM=<S,\cong,\Omega>$, если и только если для любого полуавтомата $P\in\Omega$, такого, что S^{δ} не является эквивалентным полуавтомату P^{δ}, существует тестовая последовательность, множества реакций на которую полуавтоматов S^{δ} и P^{δ} различны, т. е. различными являются множества реакций автоматов $M^{\delta\omega}{}_S$ и $M^{\delta\omega}{}_P$ (утверждение 2), и, следовательно, тест является полным относительно автоматной модели FM_{FSM}.

Для автоматной модели FM_{FSM} существуют методы построения полных те-

стов относительно FM_{FSM} для недетерминированных и детерминированных, полностью определенных и частичных автоматов-спецификаций. Наиболее хорошо развиты методы построения полных проверяющих тестов для случая, когда автомат-спецификация является приведенным детерминированным автоматом, а множество $\Omega^{\delta\omega}$ содержит все детерминированные полностью определенные автоматы с числом состояний не более некоторого заданного числа m. Рассмотрим более подробно метод построения проверяющего теста для случая, когда автомат-спецификация $M^{\delta\omega}{}_S$ является полностью определенным приведенным автоматом.

Пусть в модели $FM=<S,\cong,\Omega(m)>$ полуавтомат-спецификация является полностью определенным по входным символам и детерминированным по выходным символам, и $\Omega(m)$ есть множество полностью определенных детерминированных (по выходным символам) полуавтоматов с числом состояний не более m. Пусть, кроме того, конечный автомат $M^{\delta\omega}{}_S$ имеет n состояний, $n \le m$, и является приведенным, т. е. любые два состояния отличаются по реакции на некоторую входную последовательность. Для любых двух состояний существует входная последовательность, на которую выходные реакции в этих состояниях различны. Полный проверяющий тест для автоматной модели $<M^{\delta\omega}{}_S,\cong_{FSM},\Omega^{\delta\omega}(m)>$ можно построить W-методом или его различными модификациями [4] с использованием различающих и передаточных последовательностей. На первом шаге строятся множество различимости W, содержащее различающую последовательность для каждой пары состояний, и множество достижимости, содержащее передаточную последовательность для каждого из состояний. Тогда множество $V.W \cup VI.W \cup \ldots \cup VI^{m-n+1}.W$ является полным тестом относительно модели неисправности $<M^{\delta\omega}{}_S,\cong_{FSM},\Omega^{\delta\omega}(m)>$, следовательно, согласно утверждению 3, полным тестом относительно $<S,\cong,\Omega(m)>$. Напомним, что тестовые последовательности содержат входные символы и символ ω и подаются на тестируемый полуавтомат выше описанным образом.

Если автомат-спецификация не является приведенным, то достаточно воспользоваться его приведенной формой. Если m совпадает с числом состояний приведенной формы автомата-спецификации $M^{\delta\omega}{}_S$, то сложность построения и

общая длина полного проверяющего теста являются полиномиальными относительно *m*. Если *m* больше числа *n* состояний приведенной формы автомата $M^{\delta\omega}{}_S$, то общая длина полного проверяющего теста пропорциональна $(|I|+1)^{m-n+1}$, где *I* – входной алфавит полуавтомата-спецификации.

Если автомат-спецификация является детерминированным приведенным, но частичным, то можно воспользоваться гармонизированными идентификаторами состояний, которые используются вместо множества различимости. В примере на рис. 1б система гармонизированных идентификаторов имеет вид $H=\{H_1,H_2,H_3,H_4,H_5\}$, в которой $H_1=H_2=H_3=\{\omega\omega\}$ и $H_4=H_5=\{\omega\omega i\omega\}$. В качестве множества достижимости используем множество $V=\{\varepsilon,\omega,i,i\omega,\omega\omega\}$, где ε – пустая последовательность. В результате получим множество входных последовательностей $TS=\{ii\omega\omega,i\omega i\omega\omega,i\omega\omega\omega i\omega,\omega\omega i\omega\omega,\omega\omega\omega\omega i\omega\}$, которое является полным проверяющим тестом относительно модели неисправности $<S,\cong_q,\Omega(5)>$, в которой спецификация есть полуавтомат S на рис. 1а, а отношение конформности $\cong_q$ требует, чтобы поведение проверяемого автомата, конформного спецификации, совпадало с поведением спецификации на допустимых для спецификации входных последовательностях.

Следует отметить, что если соответствующий частичный автомат не является приведенным, то тесты получаются более длинными, чем для полностью определенного полуавтомата-спецификации, однако теоретические оценки сложности построения и общей длины полного проверяющего теста совпадают с таковыми для полностью определенных полуавтоматов.

Модель неисправности, в которой полуавтомат-спецификация может быть недетерминированным по выходам

Рассмотрим модель неисправности $FM=<S,\leq,\Omega(n)>$, в которой полуавтомат-спецификация есть полностью определенный по входным символам наблюдаемый, возможно, недетерминированный по выходам полуавтомат, причем соответствующий автомат $M^{\delta\omega}{}_S$ имеет *n* состояний. Все полуавтоматы множества Ω детерминированные, имеют не более *n* состояний, и отношение конформности является отношением редукции, т. е. множество трасс конформного полуавтомата должно содержаться в множестве трасс полуавтомата-спецификации.

Пусть автомат $M^{\delta\omega}{}_S$ обладает разделяющей последовательностью γ, т. е. для

любых двух состояний автомата множества выходных реакций на последовательность γ не пересекаются. Предположим также, что любое состояние *s* в автомате $M^{\delta\omega}{}_S$ *детерминировано достижимо* (*д-достижимо*) из начального по некоторой входной последовательности β_s, т. е. β_s переводит автомат в это состояние независимо от выходной реакции. Множество таких д-передаточных последовательностей для всех состояний назовем множеством д-достижимости V_d. Тогда множество $V_d.\gamma \cup V_d I.\gamma$ является полным тестом относительно модели неисправности $<M^{\delta\omega}{}_S, \leq_{FSM}, \Omega^{\delta\omega}(n)>$, следовательно, согласно утверждению 3, полным тестом относительно $<S, \leq, \Omega(n)>$, при условии выполнения требований таймаутов при подаче входной и наблюдении выходной последовательности.

Если полуавтомат-спецификация является наблюдаемым, но может быть частично определенным, то вместо отношения редукции рассматривается отношение квази-редукции, которое, вообще говоря, очень близко к отношению *ioco* [5]. В этом случае отмечается, что полный проверяющий тест должен быть адаптивным, и в работе [11] приведен алгоритм построения такого теста для конечно автоматной модели относительно отношений квази-эквивалентности и квази-редукции. Построенные полные тесты могут быть использованы при тестировании входо-выходных ($T_{вх}$,$T_{вых}$)-полуавтоматов. Если полуавтоматы множества Ω могут быть недетерминированными по выходам, то при подаче полного теста должно выполняться требование о «всех погодных условиях», т. е. каждая тестовая последовательность подается на тестируемый полуавтомат достаточное количество раз, чтобы пронаблюдать все выходные реакции.

Тесты, построенные автоматными методами, активно используются при тестировании телекоммуникационных протоколов, а также программного обеспечения для микропроцессоров (см., например, [7, 8, 16], где приведены примеры протоколов, в реализациях которых были найдены несоответствия спецификациям). В работе [17] рассмотрено использование полуавтоматной модели для проверки наличия состязаний в композиции SDN контроллера и переключателя.

ЗАКЛЮЧЕНИЕ

В настоящей работе проблема построения конечных тестов с гарантированной полнотой на основе входо-выходного полуавтомата для модели «черного

ящика» сведена к построению такого теста для конечно автоматной модели. Соответственно, оценки сложности для таких тестов для подходящей модели неисправности совпадают с оценками сложности для подходящих классических конечных автоматов. Поскольку для конечных автоматов адаптивность в некоторых случаях позволяет снизить сложность построения и длину проверяющего теста, в дальнейшем авторы предполагают рассмотреть адаптивные тестовые последовательности для полуавтоматов, а также выделить классы с «хорошими» оценками сложности для таких экспериментов.

Работа выполнена при поддержке Российского научного фонда, проект № 22-29-01189.

СПИСОК ЛИТЕРАТУРЫ

1. *Hennie F.C.* Fault-Detecting Experiments for Sequential Circuits // The Fifth Ann. Symp. Switching Circuit Theory and Logical Design. 1964. P. 95–110.

2. *Василевский М. П.* О распознавании неисправности автоматов // Кибернетика. 1973. № 4. С. 98–108.

3. *Bochmann G., Petrenko A.* Protocol testing: review of methods and relevance for software testing // Intern. Symp. on Software Testing and Analysis. 1994. P. 109–123.

4. *Dorofeeva R., El-Fakih K., Cavalli A., Maag S., Yevtushenko N.* FSM-based conformance testing methods: A survey annotated with experimental evaluation // Information & Software Technology. 2010. Vol. 52. No 12. P. 1286–1297.

5. *Tretmans J.* A formal approach to conformance testing // The Intern. Workshop on Protocol Test Systems. 1993. P. 257–276.

6. *Бурдонов И.Б., Косачев А.С., Кулямин В.В.* Теория соответствия для систем с блокировками и разрушением. М.: Наука, Глав. ред. физ.-мат. лит., 2008. 412 с.

7. *Kushik N., Forostyanova M., Prokopenko S., Yevtushenko N.* Studying the optimal height of the EFSM equivalent for testing telecommunication protocols // Intern. Conf. on Advances in Computing, Communication and Information Technology. 2014. P. 159–163.

8. *Жигулин М.В., Коломеец А.В., Кушик Н.Г., Шабалдин А.В.* Тестирова-

ние программной реализации протокола IRC на основе модели расширенного автомата // Известия Томского политехнического университета. 2011. Т. 318. № 5. С. 81–84.

9. *Lee D, Yannakakis M.* Principles and methods of testing finite-state machines – a survey // Proceedings of the IEEE. 1996. Vol. 84. No. 8. P. 1089–1123.

10. *Petrenko A., Y*evtushenko N. Testing from Partial Deterministic FSM Specifications // IEEE Trans. Computers. 2005. Vol. 54. No. 9. P. 1154–1165.

11. *Petrenko A., Yevtushenko N.* Conformance Tests as Checking Experiments for Partial Nondeterministic FSM // Lecture Notes in Computer Science. 2005. Vol. 3997. P. 118–133.

12. *Yevtushenko N., Burdonov I., Kossachev A.* Deriving Distinguishing Sequences for Input/Output Automata // The IEEE East-West Design & Test Symposium. 2020. P. 1–5.

13. *Бурдонов И.Б., Евтушенко Н.В., Косачев А.С.* Синтез тестов с гарантированной полнотой для входо-выходных полуавтоматов // XXIV Всероссийская научная конференция «Научный сервис в сети Интернет». 2022. С. 93–103.

14. *Гилл А.* Введение в теорию конечных автоматов. Наука, 1966. 272 с.

15. *Petrenko A., Yevtushenko N., Bochmann G.* Fault models for testing in context // Intern. Conf. on Formal Description Techniques IX. 1996. P. 163–178.

16. *Жигулин М.В.* Методы синтеза проверяющих тестов с гарантированной полнотой для контроля дискретных управляющих систем на основе временных автоматов. Дис. ... канд. тех. наук. 2012. 109 с.

17. *Vinarskii E., Lopez J., Kushik N., Yevtushenko N., Zeghlache D.* A model checking based approach for detecting sdn races // The 31st IFIP WG 6.1 Intern. Conf. on Testing Software and Systems. 2019. P. 194–211.

USING FSM-BASED STRATEGIES FOR DERIVING TESTS WITH GUARANTEED FAULT COVERAGE FOR INPUT/OUTPUT AUTOMATA

I. Burdonov[1] [0000-0001-9539-7853], N. Yevtushenko[2] [0000-0002-4006-1161],

A. Kossachev[3] [0000-0001-5316-3813]

[1, 2,3]Ivannikov Institute for system programming of the Russian Academy of Sciences, Moscow

[1]igor@ispras.ru, [2]evtushenko@ispras.ru, [3]kos@ispras.ru

Abstract

In this paper, we study the possibility of using Finite State Machine (FSM-) based methods for deriving finite test suites with guaranteed fault coverage for Input / Output automata. A method for deriving an FSM for a given automaton is proposed and it is shown that finite test suites derived for such an FSM are complete for two fault models based on Input/Output automata if they are applied within the framework of proper timeouts.

Keywords*: Input/Output automaton, Finite State machine, fault model, complete test suite*

REFERENCES

1. *Hennie F.C.* Fault-Detecting Experiments for Sequential Circuits // The Fifth Ann. Symp. Switching Circuit Theory and Logical Design. 1964. P. 95–110.

2. *Vasilevskii M.P.* Failure diagnosis of automata. translated from Kibernetika. 1973. No. 4. P. 98–108.

3. *Bochmann G.V., Petrenko A.* Protocol testing: review of methods and relevance for software testing // Intern. Symp. on Software Testing and Analysis. 1994. P. 109–123.

4. *Dorofeeva R., El-Fakih K., Cavalli A., Maag S., Yevtushenko N.* FSM-based conformance testing methods: A survey annotated with experimental evaluation // Information & Software Technology. 2010. Vol. 52. No. 12. P. 1286–1297.

5. *Tretmans J.* A formal approach to conformance testing // The Intern. Workshop on Protocol Test Systems. 1993. P. 257–276.

6. *Burdonov I.B., Kossachev A.S., Kuliamin V.V.* Conformance theory for systems with blocking and destruction. Nauka, 2008. 412 p.

7. *Kushik N., Forostyanova M., Prokopenko S., Yevtushenko N.* Studying the optimal height of the EFSM equivalent for testing telecommunication protocols // Intern. Conf. on Advances in Computing, Communication and Information Technology. 2014. P. 159–163.

8. *Zhigulin M.V., Kolomeez A.V., Kushik N.G., Shabaldin A.V.* Testing an IRC implementation using an extended FSM model // Bulletin of the Tomsk Polytechnic University. 2011. Vol. 318. No. 5. P. 81–84.

9. *Lee D, Yannakakis M.* Principles and methods of testing finite-state machines – a survey // Proceedings of the IEEE. 1996. Vol. 84. No. 8. P. 1089–1123.

10. *Petrenko A., Y*evtushenko N. Testing from Partial Deterministic FSM Specifications // IEEE Trans. Computers. 2005. Vol. 54. No. 9. P. 1154–1165.

11. *Petrenko A., Yevtushenko N.* Conformance Tests as Checking Experiments for Partial Nondeterministic FSM // Lecture Notes in Computer Science. 2005. Vol. 3997. P. 118–133.

12. *Yevtushenko N., Burdonov I., Kossachev A.* Deriving Distinguishing Sequences for Input/Output Automata // The IEEE East-West Design & Test Symposium.. 2020. P. 1–5.

13. *Yevtushenko N., Burdonov I., Kossachev A.* Deriving complete test suites for Input / Output Automata // XXIV Russian conference «Scientific Service on the Internet». 2022. P. 93–103.

14. *Gill A.* Introduction to automata theory. M.: Nauka, 1966. 272 p.

15. *Petrenko A., Yevtushenko N., Bochmann G.V.* Fault models for testing in context // Intern. Conf. on Formal Description Techniques IX. 1996. P. 163–178.

16. *Zhigulin M.V.* Timed FSM based test derivation methods for checking control systems. PhD thesis. 2012. 109 p.

17. *Vinarskii E., Lopez J., Kushik N., Yevtushenko N., Zeghlache D.* A model checking based approach for detecting sdn races // The 31st IFIP WG 6.1 Intern. Conf. on Testing Software and Systems. 2019. P. 194–211.

СВЕДЕНИЯ ОБ АВТОРАХ

БУРДОНОВ Игорь Борисович – ведущий научный сотрудник Института системного программирования им. В.П. Иванникова РАН. Сфера научных интересов – моделирование и верификация программных систем, теория графов, теория автоматов

Igor Borisovich BURDONOV *– Leading Researcher, Ivannikov Institute for System Programming of the RAS. Research interests – modeling and verification of software systems, graph theory, automata*

email: igorburdonov@yandex.ru; igor@ispras.ru

ORCID: 0000-0001-9539-7853

ЕВТУШЕНКО Нина Владимировна – ведущий научный сотрудник Института системного программирования им. В.П. Иванникова РАН. Сфера научных интересов – теория автоматов, моделирование, верификация и верификация программных систем, телекоммуникационные протоколы и сервисы.

Nina Vladimirovna YEVTUSHENKO *– leading researcher of Ivannikov Institute for System Programming of the RAS. Research interests – automata theory, modeling, verification and testing of software systems, telecommunication protocols and services*

email: nyevtush@gmail.com; evtushenko@ispras.ru

ORCID: 0000-0002-4006-1161

КОСАЧЕВ Александр Сергеевич – ведущий научный сотрудник Института системного программирования им. В.П. Иванникова РАН. Сфера научных интересов – моделирование и верификация программных систем, теория графов, теория автоматов

Alexander Sergeevich KOSSACHEV *– Leading Researcher, Ivannikov Institute for System Programming of the RAS. Research interests - modeling and verification of software systems, graph theory, automata*

email: askosachev@gmail.com; kos@ispras.ru

ORCID: 0000-0001-5316-3813

Материал поступил в редакцию 23 января 2023 года

УДК 004.891

АНАЛИЗ РАСПРЕДЕЛЕНИЯ КЛЮЧЕВЫХ ТЕРМИНОВ В НАУЧНЫХ СТАТЬЯХ

С. А. Власова[1] [0000-0003-1533-5850], Н. Е. Каленов[2] [0000-0001-5269-0988], И. Н. Соболевская[3] [0000-0002-9461-3750]

[1–3]Межведомственный суперкомпьютерный центр (МСЦ) РАН – филиал ФГУ ФНЦ Научно-исследовательский институт системных исследований (НИИСИ) РАН

[1]vlas.svetlana2013@yandex.ru, [2]nekalenov@mail.ru, [3]nik_first@mail.ru

Аннотация

Одними из основных компонентов Единого Цифрового Пространства Научных Знаний (ЕЦПНЗ) являются предметные онтологии отдельных тематических подпространств, включающие в себя основные понятия, относящиеся к данному научному направлению. Задача построения предметных онтологий на первом этапе требует формирования массива ключевых терминов в заданной области науки с последующим установлением связей между ними. Аналогичная задача стоит и при формировании энциклопедий в части определения перечня статей (слотов), определяющего их содержание. Одним из источников формирования массива ключевых терминов могут являться метаданные статей, опубликованных в ведущих научных журналах, а именно, авторские ключевые термины («ключевые слова» – в терминологии редакций журналов), сопровождающие в обязательном порядке эти статьи. Чтобы сделать заключение о возможности использования этого подхода к формированию предметных онтологий, необходимо провести предварительный анализ массива авторских ключевых терминов как с точки зрения реального соответствия основным направлениям исследований в данном разделе науки, так и с точки зрения распределения частоты встречаемости тех или иных терминов. В данной статье приведены результаты частотного анализа встречаемости авторских ключевых терминов на русском и английском языках, проведенного на основе программной обработки нескольких тысяч статей из ведущих российских журналов по математике, информатике и физике, отраженных в базе данных MathNet и на сайтах ряда издательств. Проведена

оценка соответствия распределения ключевых терминов (как словосочетаний) и отдельных слов закону Брэдфорда, выявлены ядра ключевых терминов внутри тематических направлений.

Ключевые слова: *цифровое пространство научных знаний, предметные онтологии, энциклопедические статьи, ключевые термины, метаданные статей, частотный анализ.*

ВВЕДЕНИЕ

Единое Цифровое Пространство Научных Знаний (ЕЦПНЗ) формируется как интегратор многоаспектной цифровой научной информации, достоверность которой подтверждена научным сообществом[1].

Основными целями создания ЕЦПНЗ являются предоставление различным категориям пользователей нужной им информации и обеспечение сохранности оригиналов артефактов, представляющих историческую ценность, путем создания их цифровых копий или моделей [1, 2].

Одним из основных источников контента ЕЦПНЗ является портал «Знание» [3], создаваемый на базе электронной версии Большой Российской энциклопедии с привлечением других научных энциклопедий, а также ресурсов музеев, архивов библиотек, организаций науки, образования и культуры [4].

Одной из проблем при создании научной составляющей Энциклопедии и портала «Знания» является определение перечня статей (слотов), являющихся «точками входа» в информационную систему. Эта задача, по сути, близка задаче формирования предметной онтологии, поскольку перечень статей научной энциклопедии должен тесно коррелировать с понятийной основой данного научного направления.

Таким образом, идея анализа авторских ключевых терминов с целью формирования фундамента предметной онтологии может оказаться полезной не только при проектировании ЕЦПНЗ, но и при развитии портала «Знание» и его основы – Большой российской энциклопедии.

[1] Таким подтверждением могут служить экспертные оценки, многолетнее использование результатов исследований с положительным эффектом, историческая ценность оригинала цифрового объекта и т. п.

Для получения «устойчивых» результатов, отражающих реальное распределение ключевых терминов, необходимо иметь репрезентативную выборку статей по рассматриваемому научному направлению и, соответственно, достаточно большой массив журналов, содержащих в цифровом виде информацию о ключевых терминах.

Для проведения соответствующих расчетов, касающихся русскоязычных терминов наиболее рационально было бы использовать базы данных РИНЦ или RSCI. Однако РИНЦ не дает возможности выгрузки статей в структурированном виде и закрывает возможности анализа HTML-файлов, содержащих метаданные статей, выдаваемых по запросам. За предоставление возможностей анализа массива данных РИНЦ самим пользователям администрация eLibrary требует постатейную, достаточно высокую, плату. База данных RSCI [5], которая представлена на платформе WEB of Science и содержит, как утверждается руководством РАН, наиболее важные российские журналы, в национальной подписке для российских пользователей недоступна, для работы с ней необходимо коммерческое соглашение с компанией Clarivate.

Поэтому для проведения модельных расчетов нами была выбрана отечественная система MathNet [6], которая позволяет анализировать поддерживаемую ею информацию программным образом. В дополнение к этому были проанализированы сайты журналов, не отражаемых в MathNet, на предмет возможности программного выделения ключевых терминов из метаданных опубликованных в них статей.

Для проведения анализа были разработаны структура соответствующей базы данных, специальные программные средства, обеспечивающие выделение и загрузку в базу данных необходимой информации, а также прикладные программы для анализа данных.

1. СТРУКТУРА БАЗЫ ДАННЫХ

Сформированная база данных поддерживается Microsoft SQL Server и содержит 7 видов объектов – «тематика», «журнал», «статья», ключевые термины (КТ) на русском и английском языках, ключевые слова (КС) (отдельные слова, входящие в состав терминов) на русском и английском языках. Объекты имеют следующие атрибуты.

Тематика

- Идентификатор записи
- Наименование тематики журнала
- Рубрика ГРНТИ журнала

Журнал

- Идентификатор записи
- Название журнала на русском языке
- Название журнала на английском языке

Статья

- Идентификатор записи
- Название статьи на русском языке
- Название статьи на английском языке
- Идентификатор журнала
- Год издания
- Выпуск (том, номер)
- Адрес сайта статьи

Ключевой термин на русском языке

- Идентификатор записи
- Ключевой термин на русском языке
- Идентификатор статьи
- Идентификатор журнала

Ключевой термин на английском языке

- Идентификатор записи
- Ключевой термин на английском языке
- Идентификатор статьи
- Идентификатор журнала

Ключевое слово на русском языке

- Идентификатор записи
- Ключевое слово на русском языке
- Идентификатор ключевого термина

Ключевое слово на английском языке

- Идентификатор записи
- Ключевое слово на английском языке
- Идентификатор ключевого термина

Программная оболочка системы, обеспечивающая работу с базой данных, создана на основе технологии Microsoft ASP.NET на платформе Microsoft.NET Framework в среде разработки Microsoft Visual Studio 2019.

Система представлена в свободном доступе по адресу http://dirsmsc.ru/keyterms/ и предоставляет пользователю следующие возможности.

✓ Анализ общего частотного распределения КТ и КС.

✓ Хронологический анализ распределения КТ и КС по журналам – по выбранным из ядра КТ или КС можно получить их частотное распределение по годам, а также список журналов, в которых они встречаются (с указанием количества по годам).

✓ Анализ КТ и КС, относящихся к конкретным журналам, – по выбранным журналам можно получить списки ядра КТ и КС (с указанием частоты их встречаемости).

2. ОТБОР МАТЕРИАЛА ДЛЯ ПРОВЕДЕНИЯ АНАЛИЗА

Для эксперимента были отобраны следующие журналы по математике, физике и информатике.

Математика [6]

Известия Российской академии наук. Математическая серия. Количество статей – 573 за период 2009–2021 гг.;

Математический сборник. Количество статей – 873 за период 2009–2020 гг.;

Дискретная математика. Количество статей – 249 за период 2014–2021 гг.;

Успехи математических наук. Количество статей – 251 за период 2010–

2021 гг.;

Функциональный анализ и его приложения. Количество статей – 861 за период 2003–2021 гг.;

Алгебра и анализ. Количество статей – 578 за период 2010–2021 гг.;

Алгебра и логика. Количество статей – 724 за период 2001–2020 гг.

Физика [6]

Вестник Самарского государственного технического университета. Серия «Физико-математические науки». Количество статей – 806 за период 2008–2021 гг.;

Теоретическая и математическая физика. Количество статей – 2626 за период 2002–2021 гг.

Информатика

Вычислительные методы и программирование. Количество статей – 1152 за период 2000–2021 гг. [7];

Программные продукты и системы. Количество статей – 1815 за период 2008–2021 гг. [8];

Информатика и ее приложения. Количество статей – 573 за период 2007–2021 гг. [6].

Таким образом, по математике было проанализировано более 3700 статей, в среднем, за 12-летний период; по физике – около 3500 статей, в среднем, за 16-летний период; по информатике – более 2500 статей за 13-летний период.

3. РЕЗУЛЬТАТЫ ПРОВЕДЕННОГО АНАЛИЗА

По загруженным данным для различных тематических направлений были отдельно проанализированы русскоязычные и англоязычные ключевые термины, а также входящие в них ключевые слова. Результаты анализа ключевых терминов приведены в таблицах 1, 3, 5, результаты анализа ключевых слов – в таблицах 2, 4, 6.

Таблица 1. Математика. Ключевые термины

№	Наименование	Русские	Английские
1	Всего Ключевых терминов	15949	14924
2	Различных КТ	10135	9690
3	20% из них (наиболее повторяющихся)	2027	1938
4	Всего КТ для выбранных 20% (с повторениями)	7301	6747
5	Процент повторяющихся КС из 20% от всех	45,78%	45,2%

Таблица 2. Математика. Ключевые слова

№	Наименование	Русские	Английские
1	Всего Ключевых слов	36286	34849
2	Различных КС	7358	4509
3	20% из них (наиболее повторяющихся)	1471	901
4	Всего КС для выбранных 20%	26786	27668
5	Процент повторяющихся КС из 20% от всех	73,8%	79,4%

Таблица 3. Физика. Ключевые термины

№	Наименование	Русские	Английские
1	Всего Ключевых терминов	14103	14038
2	Различных КТ	8172	8096
3	20% из них (наиболее повторяющихся)	1634	1619
4	Всего КТ для выбранных 20%	6799	6847
5	Процент повторяющихся КТ из 20% от всех	48,2%	48,8%

Таблица 4. Физика. Ключевые слова

№	Наименование	Русские	Английские
1	Всего Ключевых слов	31211	31694
2	Различных КС	6642	4216
3	20% из них (наиболее повторяющихся)	1328	843
4	Всего КС для выбранных 20%	22657	25019
5	Процент повторяющихся КС из 20% от всех	72,6%	79%

Таблица 5. Информатика. Ключевые термины

№	Наименование	Русские	Английские
1	Всего Ключевых терминов	19050	16689
2	Различных КТ	11341	9688
3	20% из них (наиболее повторяющихся)	2268	1937
4	Всего КТ для выбранных 20%	9672	8432
5	Процент повторяющихся КТ из 20% от всех	50,77%	50,52%

Таблица 6. Информатика. Ключевые слова

№	Наименование	Русские	Английские
1	Всего Ключевых слов	40913	36339
2	Различных КС	9578	5425
3	20% из них (наиболее повторяющихся)	1914	1085
4	Всего КС для выбранных 20%	29774	28406
5	Процент повторяющихся КС из 20% от всех	72,77%	78,17%

Графики распределения частоты встречаемости русских ключевых терминов и русских ключевых слов в математических журналах представлены на рис. 1 и 2 соответственно.

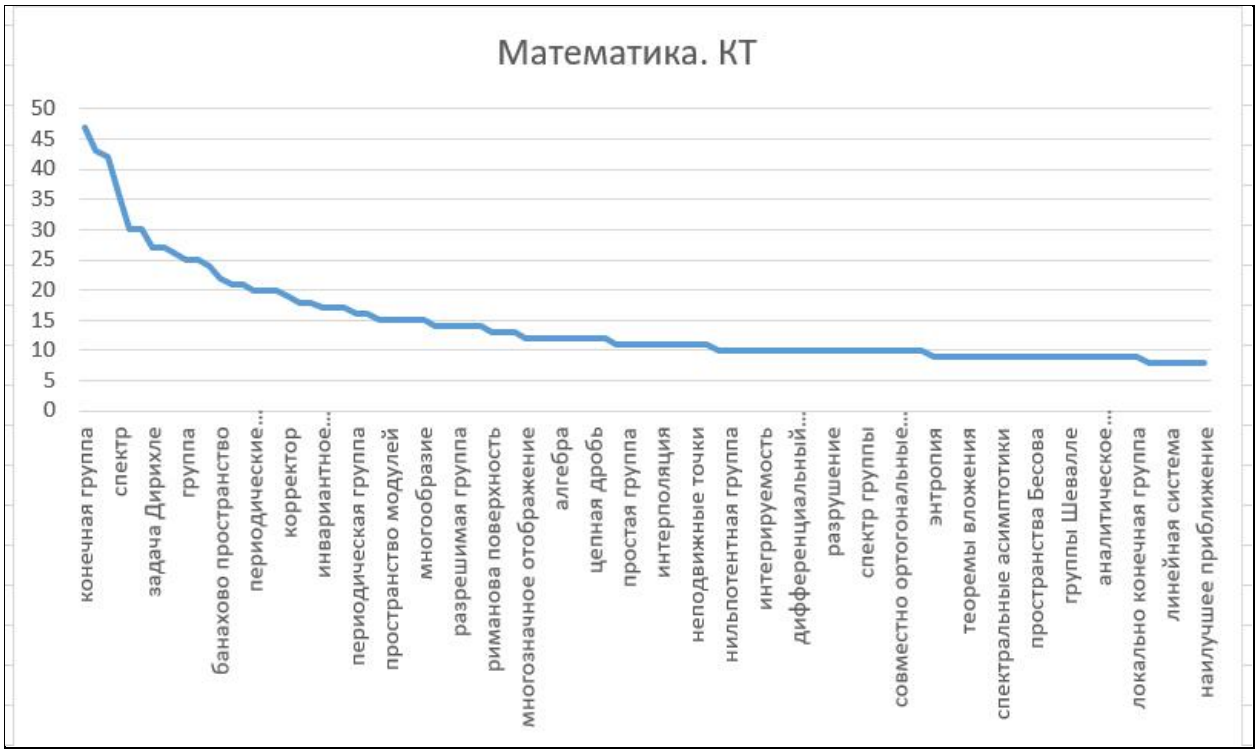


Рис. 1. Распределение русских КТ в математических журналах

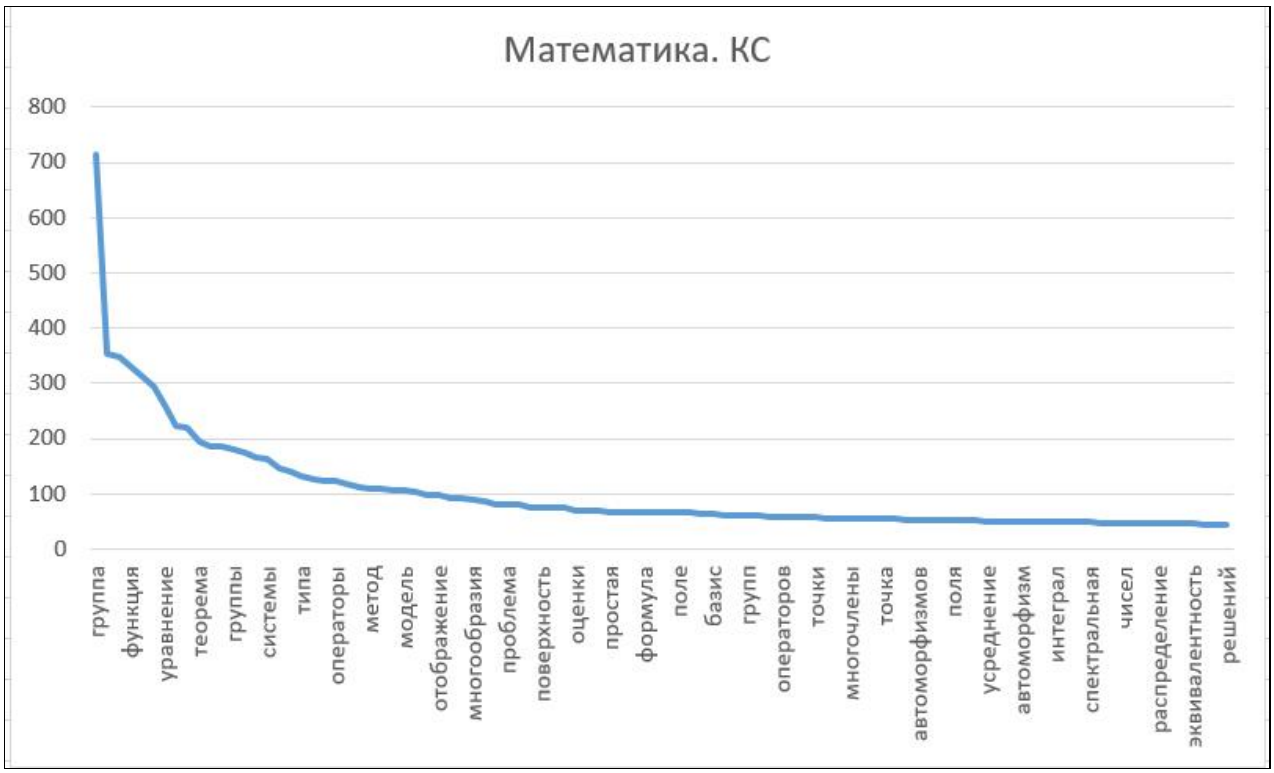


Рис. 2. Распределение русских КС в математических журналах

Если мы проанализируем список ключевых терминов, выделенных в русскоязычных журналах по информатике (его фрагмент приведен в таблице 7), то увидим, что в нем встречаются термин «параллельные алгоритмы» 29 раз, «параллельный алгоритм» 22 раза. Аналогично, в списке англоязычных терминов по информатике КТ «parallel algorithms» встречается 23 раза, а КТ «parallel algorithm» – 21 раз.

Если исключить из рассмотрения такие общие понятия, как «алгоритм», «вычисления», «компьютеры» и т. п., то, анализируя ядро перечня КТ по информатике, можно сделать вывод, что к наиболее актуальным проблемам относятся направления, связанные с:

– моделированием (в списке из 200 наиболее часто используемых КТ термины, связанные с этой проблемой, встречаются 436 раз (математическое моделирование – 104 раза, моделирование – 96 раз, численное моделирование – 46 раз, модель – 45 раз, имитационное моделирование – 44 раза, математическая модель – 32 раза, компьютерное моделирование – 22 раза, модель данных – 10 раз, суперкомпьютерное моделирование – 10 раз, информационная модель – 9 раз, имитационная модель – 9 раз, аналитическое моделирование – 9 раз);

– параллельными вычислениями (в списке из 200 наиболее часто используемых КТ термины, связанные с этой проблемой, встречаются 221 раз: параллельные вычисления – 132 раза, параллельные алгоритмы – 51 раз, параллельное программирование – 38 раз);

– оптимизацией в списке из 200 наиболее часто используемых КТ термины, связанные с этой проблемой, встречаются 68 раз (оптимизация – 50 раз, глобальная оптимизация – 10 раз, многокритериальная оптимизация – 8 раз).

Таблица 7. Фрагмент рейтингового списка КТ по информатике

КТ информатика	Частота встречаемости
Параллельные вычисления	132
Математическое моделирование	104
Численные методы	97
Моделирование	96
Оптимизация	50
Алгоритм	47

Высокопроизводительные вычисления	46
Программный комплекс	46
Численное моделирование	46
Модель	45
Нейронные сети	44
Имитационное моделирование	44
Прогнозирование	43
Информационная система	41
Параллельное программирование	38
Информационная безопасность	37
Управление	36
Обратные задачи	34
Программное обеспечение	33
Принятие решений	32
Математическая модель	32
Краевые задачи	31
База знаний	31
Система массового обслуживания	31
Автоматизация	30
Машинное обучение	30
Метод конечных элементов	30
Искусственный интеллект	30
Параллельные алгоритмы	29
Мониторинг	27
Визуализация	27
Генетический алгоритм	27
Устойчивость	27
Численный анализ	26
Информационные технологии	26
Кластеризация	26
Надежность	26
Суперкомпьютер	26

САПР	25
Нечеткая логика	25
Верификация	25
Распределенные вычисления	24
Экспертная система	24
Обыкновенные дифференциальные уравнения	23
MPI[2]	23
Компьютерное моделирование	22
Итерационные методы	22
Параллельный алгоритм	22
Сходимость	22
Эффективность	22

ЗАКЛЮЧЕНИЕ

Результаты анализа показывают, что распределение ключевых терминов в том виде, как они представлены авторами, достаточно далеко от распределения Брэдфорда, в то время как распределение ключевых слов вполне ему соответствует. Более подробный анализ рейтингового списка ключевых терминов объясняет причину этого, которая в значительной степени обусловлена разной последовательностью одних и тех же слов, входящих в состав ключевого термина.

Очевидно, что для более точной картины при обработке КТ необходимо применять методы лингвистического анализа, что на данном этапе в нашу задачу не входило. Однако сформированная база данных и простой «ручной» анализ полученного «ядра» КТ позволяют сформировать список наиболее значимых терминов для последующего их включения в Единое цифровое пространство научных знаний [9].

Разработанные методика и программная оболочка позволяют проводить анализ динамики развития той или иной области науки, а также могут служить инструментом для развития и корректировки политематических и специальных

[2] Message Passing Interface

научных энциклопедий. Сравнение приведенного в табл. 7 списка из 50-ти наиболее употребительных авторских ключевых терминов с электронной версией Большой российской энциклопедии показало, что в ней отсутствуют статьи, посвященные таким терминам, как «высокопроизводительные вычисления», «имитационное моделирование», «обратные задачи», «генетический алгоритм», «MPI» и др. В БРЭ отсутствуют лидирующие в рейтинге авторских ключевых терминов «параллельный алгоритм» и «параллельные вычисления», но присутствует термин «параллельное программирование», который не используют авторы статей. Вместо распространенного термина «машинное обучение» (26-е место в рейтинге) в БРЭ приведен термин «программированное обучение».

В качестве следующего шага исследований в данном направлении планируется использовать сформированную базу данных для анализа динамики изменения состава «ядра» ключевых терминов, что представляет интерес для задач наукометрии, характеризуя, в определенной степени, динамику развития отдельных областей рассматриваемых наук.

Работа выполнена в МСЦ РАН в рамках государственного задания по теме FNEF-2023-0014.

СПИСОК ЛИТЕРАТУРЫ

1. *Антопольский А.Б., Каленов Н.Е., Серебряков В.А., Сотников А.Н.* О едином цифровом пространстве научных знаний // Вестник Российской академии наук. 2019. Т. 89 (7). С. 728–735.
URL: https://doi.org/ 10.31857/S0869-5873897728-735.

2. *Савин Г.И.* Единое цифровое пространство научных знаний: цели и задачи // Информационные ресурсы России. 2020. № 5. С. 3–5.
URL: https://doi.org/ 10.51218/0204-3653-2020-5-3-5.

3. Большая российская энциклопедия. URL: https://bigenc.ru/ (дата обращения: 22.12.2022).

4. *Kalenov N., Savin G., Sotnikov A.* Fundamentals of Common Digital Space of Scientific Knowledge Building // CEUR Workshop Proceedings (CEUR-WS.org). 2021. Vol. 2990. P. 93–99. URL: https://doi.org/10.51218/1613-0073-2990-93-99

5. *Михайлов О.В.* Новая платформа журналов RSCI в WEB of Science Вестник

Российской академии наук. 2017. Т. 87. № 2. С. 177–180.

6. Общероссийский портал Math-Net.ru. URL: http://www.mathnet.ru/ (дата обращения: 22.12.2022).

7. Вычислительные методы и программирование. URL: https://num-meth.ru/index.php/journal/issue/archive (дата обращения: 22.12.2022).

8. Программные продукты и системы. URL: http://www.swsys.ru/index.php?page=10&lang= (дата обращения: 22.12.2022)

9. *Власова С.А., Каленов Н.Е., Сотников А.Н.* Web-ориентированная система формирования контента единого цифрового пространства научных знаний // Программные продукты и системы. 2020. № 3. С. 365–374. URL: https://doi.org/10.15827/0236-235X.131.365-374.

ANALYSIS OF THE DISTRIBUTION OF KEY TERMS IN SCIENTIFIC ARTICLES

S. A. Vlasova[1] [0000-0003-1533-5850], N. E. Kalenov[2] [0000-0001-5269-0988],
I. N. Sobolevskaya[3] [0000-0002-9461-3750]

[1-3]Joint Supercomputer Center of the Russian Academy of Sciences – JSC

[1]vlas.svetlana2013@yandex.ru, [2]nekalenov@mail.ru, [3]nik_first@mail.ru

Abstract

One of the Common Digital Space of Scientific Knowledge (CDSSK) main components are the subject ontologies of individual thematic subspaces, which include the basic concepts related to this scientific area. The constructing subject ontologies task at the initial phase requires the array of key terms formation in a given scientific are with the subsequent establishment of links between them. A similar task is in the encyclopedias formation in terms of the articles (slots) list generating that determines their content. One of the sources for the formation of the key terms array can be the metadata of articles published in the leading scientific journals. Namely, the author's key terms ("keywords" in the terminology of the journals editors) quoted by the article. To make a conclusion about the possibility of using this approach to the subject ontologies formation, it is necessary to conduct the author's key terms array preanalysis,

both in terms of real correspondence to the main areas of research in this science branch and in terms of the distribution of the certain terms occurrence frequency. This article presents the results of the occurrence frequency analysis of the author's key terms in Russian and English, carried out on the software processing basis of several thousand articles from leading Russian journals in mathematics, computer science and physics, reflected in the MathNet database. An assessment was made of the distribution of key terms correspondence (as phrases) and individual words to the Bradford's law, and the key terms cores within the thematic direction were identified.

Keywords: *digital space of scientific knowledge, subject ontologies, encyclopedia articles, key terms, article metadata, frequency analysis.*

REFERENCES

1. *Antopol'skij A.B., Kalenov N.E., Serebryakov V.A., Sotnikov A.N.* O edinom cifrovom prostranstve nauchnyh znanij // Vestnik Rossijskoj akademii nauk, 2019. V. 89 (7). S. 728–735. URL: https://doi.org/10.31857/S0869-5873897728-735.

2. *Savin G.I.* Edinoe cifrovoe prostranstvo nauchnyh znanij: celi i zadachi // Informacionnye resursy Rossii. 2020. № 5. S. 3–5.
URL: https://doi.org/ 10.51218/0204-3653-2020-5-3-5.

3. Bol'shaya rossijskaya enciklopediya. URL: https://bigenc.ru/ (accessed 22 December 2022).

4. *Kalenov N., Savin G., Sotnikov A.* Fundamentals of Common Digital Space of Scientific Knowledge Building // CEUR Workshop Proceedings (CEUR-WS.org). 2021. V. 2990. P. 93–99. https://doi.org/10.51218/1613-0073-2990-93-99.

5. *Mikhailov O.V.* Novaya platforma zhurnalov RSCI on WEB of Science // Vestnik Rossiiskoi akademii nauk Вестник. 2017. V. 87. № 2. S. 177–180.

6. Obshcherossijskij portal Math-Net.ru. URL: http://www.mathnet.ru/ (accessed 22 December 2022).

7. Vychislitel'nye metody i programmirovanie.
URL: https://num-meth.ru/index.php/journal/issue/archive (accessed 22 December 2022).

8. Programmnye produkty i sistemy.
URL: http://www.swsys.ru/index.php?page=10&lang= (accessed 22 December 2022).

9. *Vlasova S.A., Kalenov N.E., Sotnikov A.N.* Web-orientirovannaya sistema formirovaniya kontenta edinogo cifrovogo prostranstva nauchnyh znanij // Programmnye produkty i sistemy. 2020. № 3. S. 365–374.
URL: https://doi.org/10.15827/0236-235X.131.365-374.

СВЕДЕНИЯ ОБ АВТОРАХ

ВЛАСОВА Светлана Александровна – ведущий научный сотрудник Межведомственного Суперкомпьютерного Центра РАН – филиала Федерального государственного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук», кандидат технических наук.

Svetlana Aleksandrovna VLASOVA – Leading Researcher of Joint Super Computer Center of the Russian Academy of Sciences – Branch of Federal State Institution «Scientific Research Institute for System Analysis of the Russian Academy of Sciences», Candidate of Technical Sciences

email: vlas.svetlana2013@yandex.ru;
ORCID: 0000-0003-1533-5850.

КАЛЕНОВ Николай Евгеньевич – главный научный сотрудник Межведомственного Суперкомпьютерного Центра РАН – филиала Федерального государственного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук», доктор технических наук, профессор.

Nikolay Evgenievich KALENOV – Chief Researcher of Joint Super Computer Center of the Russian Academy of Sciences – Branch of Federal State Institution «Scientific Research Institute for System Analysis of the Russian Academy of Sciences», Doctor of Technical Sciences, Professor.

email: nekalenov@mail.ru;
ORCID: 0000-0001-5269-0988.

Соболевская Ирина Николаевна – старший научный

сотрудник Межведомственного Суперкомпьютерного Центра РАН – филиала Федерального государственного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук», кандидат физико-математических наук.

Sobolevskaya Irina Nikolaevna – higher senior officer of Joint Super Computer Center of the Russian Academy of Sciences – Branch of Federal State Institution «Scientific Research Institute for System Analysis of the Russian Academy of Sciences», Candidate of Physics and Math Sciences.

email: nik_first@mail.ru;
ORCID: 0000-0002-9461-3750

Материал поступил в редакцию 3 января 2023 года

УДК 004.021; 004.42

КАК ЭМБЕДДИНГИ ИМЕН СУЩНОСТЕЙ ВЛИЯЮТ НА КАЧЕСТВО ВЫРАВНИВАНИЯ СУЩНОСТЕЙ

Д. И. Гусев[1] [0000-0001-9636-2783], З. В. Апанович[2] [0000-0002-5767-284X]

[1]*Новосибирский государственный университет,* ул. Пирогова, 1 Новосибирск, 630090
[2]*Институт систем информатики им. А.П. Ершова Сибирского отделения Российской академии наук, пр. Академика Лаврентьева, 6, Новосибирск, 630090*

[1]d.gusev1@g.nsu.ru, [2]apanovich_09@mail.ru

Аннотация

Алгоритмы установления соответствия между сущностями осуществляют поиск эквивалентных сущностей в разноязычных графах знаний. Данная проблема возникает, как правило, при интеграции разноязычных графов знаний. В настоящее время решение этой проблемы становится весьма актуальным для практического решения проблем импортозамещения, например, чтобы найти информацию о лекарствах, выпускаемых в разных странах под разными названиями, или же решить проблему поиска эквивалентных запчастей.

В настоящее время известно несколько библиотек с открытым кодом, которые объединяют известные алгоритмы выравнивания сущностей, а также тестовые наборы данных для различных языков. В данной работе описан русско-английский набор данных для экспериментов с нескольким популярными алгоритмами выравнивания сущностей. Особое внимание уделено методам генерации векторных представлений для имен сущностей. В частности, рассмотрены комбинации различных методов генерации векторных представлений (эмбеддингов) имен сущностей с известными алгоритмами выравнивания сущностей. Таблицы с результатами экспериментов дополнены визуализациями.

Ключевые слова*: разноязычные графы знаний, идентификация сущностей, cross-lingual entity alignment, knowledge graphs, relational embeddings, name embeddings.*

ВВЕДЕНИЕ

В последние годы графы знаний (ГЗ) используются все в большем количестве предметных областей, и все большее количество приложений использует этот тип представления для хранения данных без потери их семантики. Чем мощнее граф знаний, тем выше качество приложений, на них базирующихся. Поэтому весьма актуальной является задача интеграции различных графов знаний, а в основе такой интеграции находится решение задачи слияния информации из разных графов знаний об одном и том же объекте реального мира. Данная задача известна под такими названиями, как *сопоставление сущностей*, *выравнивание сущностей*, *идентификация сущностей* и др. В последние несколько лет возрос интерес к интеграции разноязычных графов знаний, поэтому весьма актуальной является задача связывания информации об одних и тех же объектах реального мира, описанных в разноязычных графах знаний. Разные языковые версии графов знаний обладают, с одной стороны, свойством взаимодополнительности, а с другой стороны, каждая языковая версия содержит более точную и полную информацию об объектах, характерных для конкретного языка. Например, русскоязычная версия DBpedia содержит более полную и корректную информацию об объектах, расположенных на территории России.

Графы знаний хранят факты в виде реляционных и литеральных триплет. Реляционные триплеты изображают отношение между двумя объектами реального мира и имеют формат *tr_r* = (*субъектная сущность, отношение, объектная сущность*). Литеральные триплеты хранят информацию об атрибутах объектов реального мира и имеют формат *tr_l* = (*субъектная сущность, атрибут, литеральное значение*). При визуализации литеральные значения принято изображать прямоугольниками, а сущности или объекты реального мира – овалами.

Например, на рис. 1 показаны фрагменты из англоязычной и русскоязычной версий DBpedia. Примером реляционной триплеты является триплета (*Сталкер_(фильм), режиссер, Андрей_Тарковский*), а примером литеральной триплеты является триплета (*Сталкер_(фильм), длительность, 163 минуты*). Красными линиями изображены отношения *owl:sameAs*, имеющиеся между сущностями в русскоязычном и англоязычном графах знаний. Можно видеть, что англоязычной

сущности *Stalker (1979 film)* в англоязычном графе знаний соответствует русскоязычная сущность *Сталкер_(фильм),* англоязычному отношению *dbo:director* – русскоязычное отношение *режиссер*, а англоязычной сущности *Andrei Tarkovsky* – русскоязычная сущность *Андрей Тарковский.* Понятно, что англоязычный и русскоязычный списки актеров, сыгравших роли в этом фильме, должны бы совпадать. Однако в реальных графах знаний наблюдаются некоторые различия в описаниях смежных сущностей. Так, длительность фильма в англоязычной версии указана в секундах, а в русскоязычной версии – в минутах, в русскоязычной версии указаны братья Стругацкие в качестве авторов сценария, а в англоязычной версии этой информации нет. Понятно, что наиболее полное описание сущности можно получить объединением всех триплет, описывающих одну и ту же сущность. Но для решения этой задачи должно быть правильно установлено соответствие между сущностями. Эта задача и носит название *выравнивание сущностей*.

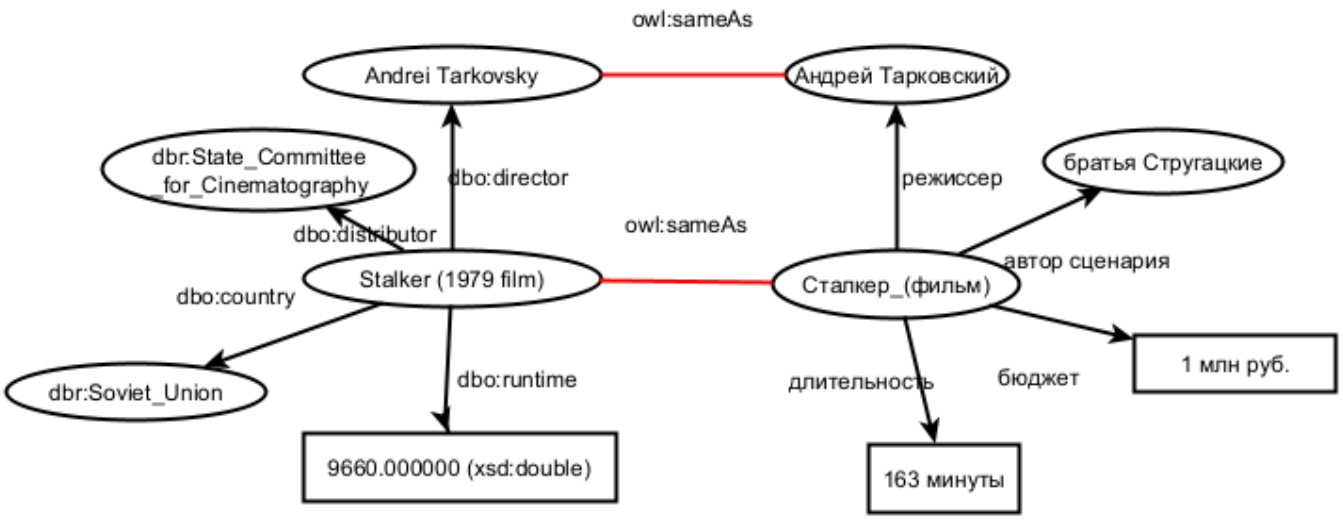


Рис. 1. Соответствие между англоязычными и русскоязычными сущностями.

В последние несколько лет получили распространение методы установления соответствия между сущностями различных графов знаний, использующие так называемые «эмбеддинги» (embeddings), векторные представления заданной размерности для сущностей и отношений графов знаний. Достоинствами подхода на основе эмбеддингов являются высокая масштабируемость и небольшие усилия при подготовке обучающих выборок.

Следует сказать, что создание новых методов основано на интуиции разработчиков, эвристиках и экспериментах проб и ошибок. Поэтому весьма важным является создание общей основы для понимания разнообразных методов. В настоящее время такую общую основу составляют результаты тестирования различных алгоритмов на едином наборе данных. В работе [1] представлена библиотека OpenEA, содержащая несколько десятков алгоритмов ЕА на основе различных стратегий построения векторных представлений, а также результаты экспериментов с этими векторными представлениями на тестовой выборке, содержащей англо-немецкие, англо-французские и англо-китайские данные.

Понятно, что русскоязычному пользователю интересны, прежде всего, эксперименты, использующие русскоязычные данные. Во-первых, такие данные проще интерпретировать, во-вторых, известно, что различные языковые версии графов знаний обладают свойством «смещенности», то есть одни и те же алгоритмы могут давать разные результаты на разных версиях графов знаний из-за различной структуры этих графов.

В работе [2] описан русско-английский набор данных для экспериментов с алгоритмами кросс-языкового выравнивания сущностей. К удивлению авторов, алгоритмы, выдававшие наилучшие результаты на стандартных разноязычных наборах данных, выдавали весьма посредственные результаты на русско-английском наборе данных. Этот вопрос потребовал дополнительного изучения, и в данной работе представлены эксперименты с алгоритмами выравнивания сущностей разного типа на англо-русской обучающей выборке. Рассмотрены различные способы построения векторных представлений имен сущностей, а также возможные комбинации этих методов с методами построения векторных представлений сущностей на основе реляционных триплет.

1. ГРУППЫ АЛГОРИТМОВ СОПОСТАВЛЕНИЯ СУЩНОСТЕЙ НА ОСНОВЕ ВЕКТОРНЫХ ПРЕДСТАВЛЕНИЙ (EMBEDDINGS)

Большинство методов выравнивания сущностей на основе векторных представлений сводится к двум шагам:

- Генерация векторных представлений для сущностей и отношений;

- Отображение этих векторных представлений в единое векторное пространство при помощи предварительно выровненных сущностей (seed alignments) или в различные векторные пространства.

В первом случае вопрос, являются ли две сущности из разных графов эквивалентными (соответствующими одному и тому же объекту реального мира), решается при помощи сравнения их векторов, например, вычислением евклидова расстояния или косинусной близости. При отображении сущностей двух графов знаний в разные векторные пространства нужно также находить матрицу соответствия между векторами этих двух пространств.

Современные решения выравнивания сущностей опираются, в основном, на структурную информацию в графах знаний, то есть реляционные триплеты. Основу этих методов составляет предположение о том, что эквивалентные сущности должны иметь сходные графовые окрестности. При появлении методов выравнивания сущностей преобладал так называемый триплетно-трансляционный подход, который рассматривал вектор, представляющий отношение между двумя сущностями, как вектор сдвига вектора одной сущности относительно вектора второй сущности. Одним из лучших представителей триплетно-трансляционного подхода является MultiKE (Multi-view Knowledge Graph Embedding) [3]. MultiKE строит три типа векторных представлений для каждой сущности, используя так называемые «виды» (views):

- вид, зависящий от названия сущности,
- «реляционный вид», конструируемый по реляционным триплетам каждой субъектной сущности,
- «атрибутный вид», создаваемый по литеральным триплетам субъектной сущности.

Каждый из «видов» строится по собственному алгоритму. Например, для каждого слова из названия сущности находится вектор, полученный с помощью word2vec [4], а если такого не существует, то вектор слова получается с помощью суммирования векторов символов, полученных с помощью алгоритма character embedding. Векторы слов суммируются, и получается вектор названия, который непосредственно участвует в обучении модели.

Для построения реляционного вида используется модель TransE [5], где отношение между двумя сущностями интерпретируется как вектор сдвига между сущностью-субъектом и сущностью-объектом реляционной триплеты. Наконец, атрибутные виды строятся на основе литеральных триплет, в которых данная сущность является субъектом. Для построения атрибутных представлений используются сверточные нейронные сети. Окончательное векторное представление сущности может быть получено при помощи разных способов комбинирования упомянутых трех видов.

В последние годы чрезвычайно популярными стали подходы построения векторных представлений сущностей на основе графовых сверточных сетей. Эти методы выдают очень неплохие результаты, но их основными недостатками являются чрезвычайная сложность, значительное время вычислений и плохая интерпретируемость. Представителем этого подхода является RDGCN (Relation-aware Dual-Graph Convolutional Network) [6]. Подход RDGCN использует для построения векторных представлений не только структуру исходных графов знаний (primal entity graph), но и вспомогательные графы, двойственные по отношению к исходным графам (dual relation graph), вершинами которых являются ребра исходных графов. Для осуществления взаимодействия между исходными графами знаний и двойственными реляционными графами используется механизм графовых сетей внимания (Graph Attention Networks, GAT) [7]. Результирующие векторные представления исходных графов затем подаются в графовые сверточные сети (Graph Convolutional networks, GCN) [8] для извлечения информации о структуре окружений вершин.

Совсем недавно появился чрезвычайно простой подход к выравниванию сущностей под названием SEU (Simple but Effective Unsupervised EA method) [9], не использующий нейронные сети. Основная идея SEU состоит в сведении задачи выравнивания сущностей к давно известной задаче назначения, для которой существует хорошо известный венгерский алгоритм решения. Основным предположением этого подхода является то, что матрицы смежностей двух графов знаний являются изоморфными. В этом случае матрица смежности исходного графа может быть преобразована в матрицу смежности второго графа посредством переупорядочения строк или столбцов.

Тем не менее, большинство недавних исследований указывает на то, что современные методы выравнивания сущностей не способны выдавать удовлетворительные результаты только на основании реляционных триплет, если набор данных имеет распределение степеней сущностей, близкое к реальным КГ. В частности, известно, что примерно половина сущностей в реальных КГ связана с менее чем тремя другими сущностями [9].

Это наблюдение делает важным использование дополнительной информации, такой как имена сущностей и комбинирование информации об именах сущностей со структурной информацией. Названия сущностей необходимо привести к общему языку, а затем сравнить. Возможны два базовых подхода для сравнения имен сущностей: подход на основе строкового сходства и подход на основе семантического сходства. Методы семантического сходства можно разбить на две группы: генерация векторных представлений на основе отдельных слов (модели word2vec, GloVe [10]). В силу ограниченности используемых словарей, часто возникает ситуация, что нужное слово отсутствует в используемом словаре, и в этом случае векторное представление слова строится на основе литер, входящих в его состав (модели fastText [11], name-BERT [12]).

2. РУССКО-АНГЛИЙСКИЙ НАБОР ДАННЫХ И МЕТРИКИ ДЛЯ ОЦЕНКИ КАЧЕСТВА АЛГОРИТМОВ ВЫРАВНИВАНИЯ СУЩНОСТЕЙ

Современные графы знаний имеют значительные размеры, поэтому вместо полномасштабных экспериментов по установлению соответствия между сущностями из разных графов знаний осуществляются эксперименты на выборках ограниченного размера. В настоящее время принято экспериментировать с выборками, содержащими 15 000 и 100 000 соответствий между сущностями из двух графов знаний. Наибольшее распространение получил набор данных DBP15K [1], который содержит по 15 000 пар сущностей, связанных отношениями *owl:sameAs* из разных языковых версий DBpedia, для таких пар языков, как англо-китайский, англо-французский и англо-немецкий. В [1] также описан итеративный алгоритм построения разноязычной выборки на основе степеней сущностей IDS (Iterative Degree-based Sampling), в которой распределение степеней сущностей близко к распределениям степеней в реальных графах знаний. Основная идея алгоритма IDS выглядит следующим образом [1].

В первую очередь удаляются сущности, которые не имеют связей *owl:sameAs* между двумя графами знаний. Пусть $P(x)$ — это доля сущностей, имеющих степень x в текущем графе знаний, а $Q(x)$ – доля сущностей, имеющих степень x в исходном графе знаний. Для оценки различия между распределениями степеней сущностей в двух наборах данных используется дивергенция Иенсена–Шэннона [13]. Доля сущностей, имеющих степень x в текущем наборе данных $P(x)$, не может быть равной доле сущностей, имеющих степень x в исходном наборе данных $Q(x)$. Поэтому вычисляется количество сущностей, которые надо удалить на одном шаге алгоритма, по формуле $dsize(x, \mu)=\mu(1+P(x)–Q(x))$, где μ – это базовый размер шага. Чтобы сбалансировать эффективность и безопасность удаления, устанавливается μ=100 при генерации тестовой выборки из 15 тысяч пар разноязычных триплет и μ=500 при генерации тестовой выборки из 100 тысяч пар разноязычных триплет. Затем при помощи алгоритма PageRank вычисляются сущности, которые реально будут удалены. Сущности, удаленные из одного графа знаний, удаляются и в другом графе знаний. Приемлемым значением дивергенции Иенсена–Шэннона считается значение 5%.

Обычно строятся две версии набора данных. Версия 1 (V1) получается путем прямого использования алгоритма IDS. Для версии 2 (V2) сначала случайным образом удаляются объекты с низкими степенями (d≤5) в графе знаний-источнике, чтобы удвоить среднюю степень, и затем выполняется IDS для соответствия новому графу знаний. В результате набор данных версии V2 вдвое плотнее, чем версии V1, и более похож на реально существующие наборы данных. В литературе по выравниванию графов знаний наиболее популярными являются немецко-английский, французско-английский и китайско-английский тестовые наборы.

Принимая во внимание то, что каждая языковая версия графа знаний имеет свою собственную структуру, отличную от других графов знаний, а также то, что данные, полученные для русскоязычного графа знаний проще интерпретировать, нами был сгенерирован русско-английский набор тестовых данных на основе русскоязычной и англоязычной версий DBpedia [2]. Использовался набор данных англоязычной и русскоязычной DBpedia за 2016 год (DBpedia 2016-10, https://wiki.dbpedia.org/downloads-2016-10).

Набор DBP-15K EN-RU (V1, V2) сгенерирован на основе алгоритма IDS и доступен для свободного скачивания (https://www.dropbox.com/sh/4oh3nkzwdr1w4dv/AACZ4v8jCdR7Y4mDtS654Bega?dl=0).

Для анализа качества работы различных алгоритмов выравнивания сущностей на основе эмбеддингов принято использовать метрики hits@*k* и среднеобратный ранг (Mean reciprocal rank, MRR). Метрика hits@*k*=*n*% означает, что для *n* процентов объектов из одного графа знаний эквивалентный объект из второго графа знаний находится среди ближайший *k* соседей в векторном пространстве. Очевидно, самой показательной считается метрика hits@1, так как эта метрика соответствует алгоритму, который самостоятельно строит правильные отношения *owl:sameAs* между сущностями. Среднеобратный ранг определяется как среднее значение обратных рангов по всем запросам. Обратный ранг в данном случае означает обратное число номера (ранга) первого правильного ответа в списке откликов.

3. ВЛИЯНИЕ ПЕРЕВОДА ЛИТЕРАЛОВ НА КАЧЕСТВО АЛГОРИТМОВ ВЫРАВНИВАНИЯ СУЩНОСТЕЙ

Первая группа экспериментов с различными методами выравнивания сущностей на англо-русском наборе данных описана в [2]. Для этих экспериментов использовались алгоритмы из библиотеки OpenEA, которые применялись к русско-английской выборке. К удивлению авторов оказалось, что наилучшие результаты выдавали такие методы, как BootEA[14] и RSNA[15], в то время как такие методы, как MultiKE и RDGCN, выдававшие наилучшие результаты на англо-французских и англо-немецких данных, давали весьма посредственные результаты. Например, метод MultiKE выдавал оценку hits@1, равную всего 37.344, в то время как этот же алгоритм на англо-французской выборке давал hits@1, равный 74.133. Аналогично метод RDGCN, который выдавал hits@1, равный 77,019 на англо-французской выборке выдавал оценку hits@1, равную всего 43,256 на русско-английских данных.

При более внимательно изучении было замечено, что наибольшее ухудшение качества результатов наблюдалось на методах выравнивания сущностей, которые при построении векторных представлений сущностей использовали не только информацию о реляционной структуре графов знаний, но и информацию

о литералах, в частности, об именах сущностей. Первое предположение о причинах неудачи было связано с тем, что английский и русский языки используют разные алфавиты, поэтому векторные представления литералов попадают в разные векторные пространства.

Для решения указанной проблемы нами был разработан инструмент автоматического перевода на основе Google Translate API. На вход программы подавались язык, с которого будет осуществлен перевод, имена сущностей и литералы. Результат перевода литералов передавался в метод формирования векторного представления.

Для сравнения методов генерации векторных представлений имен сущностей без перевода и с применением предварительного перевода были построены визуализации этих векторных представлений. Эти визуализации показаны на рисунках 2–6. В качестве инструмента снижения размерности использовался метод t-SNE [16] .

На представленных изображениях английские имена сущностей имеют синий цвет, русские – красный. Это позволяет оценить эффективность метода генерации векторных представлений. Высокая степень наложения цветов говорит о том, что семантически связанные данные, представленные на разных языках, имеют сходные векторные представления. Наличие пятен одного цвета говорит о том, что в указанной области расположены сущности из одного графа знаний, а эквивалентные сущности из другого графа знаний находятся на значительном расстоянии. На Рис. 2 показаны векторные представления (эмбеддинги) для имен сущностей из набора данных EN_RU_15K_V1, сгенерированные при помощи word2vec с оригинальными настройками MultiKE. Можно видеть, что названия сущностей из англоязычного и русскоязычного наборов данных почти не пересекаются. Пересечение наблюдается только там, где сущности из двух наборов данных имеют одинаковые англоязычные названия (например, названия музыкальных альбомов или песен).

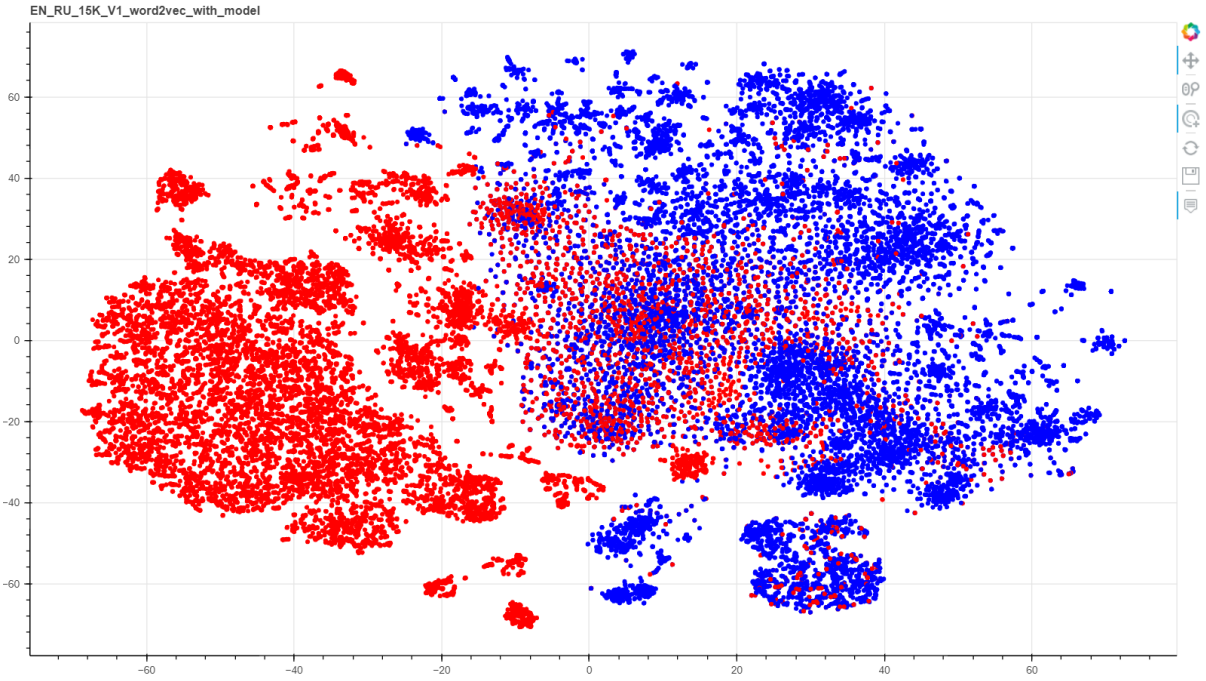


Рис. 2. Эмбеддинги имен сущностей, из набора данных EN_RU_15K_V1, сгенерированные word2vec с оригинальными настройками MultiKE (без перевода).

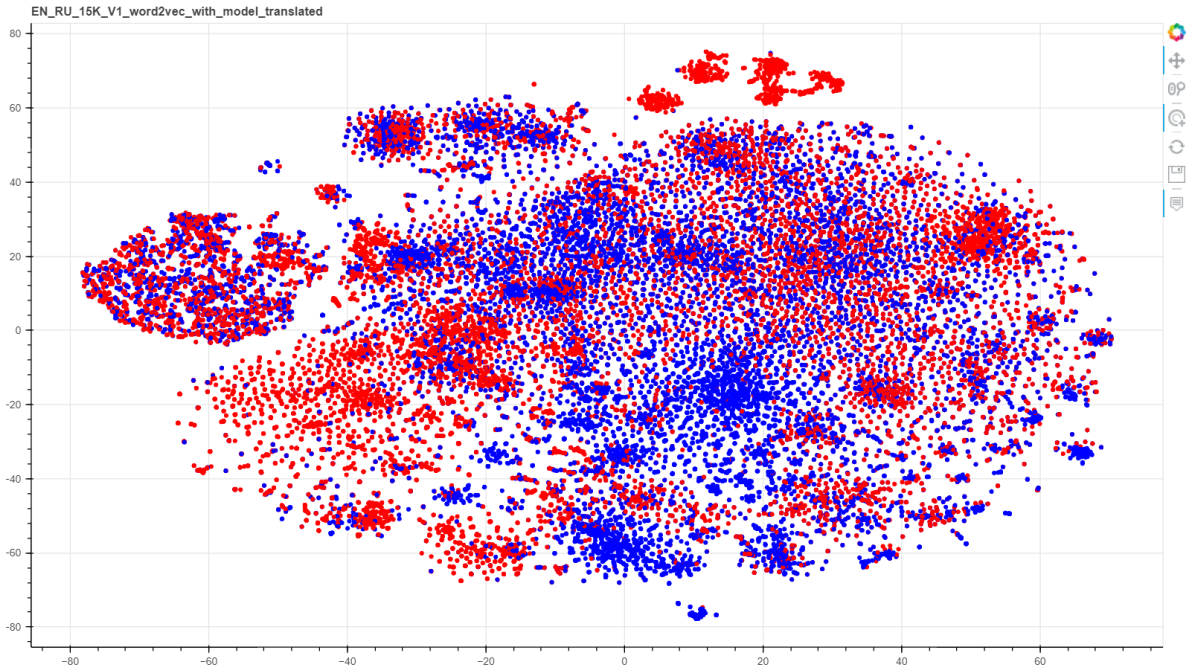


Рис. 3. Эмбеддинги имен сущностей, из набора данных EN_RU_15K_V1, сгенерированные word2vec с оригинальными настройками MultiKE (с переводом).

На рис. 3 показаны векторные представления для имен сущностей из этого же набора данных, сгенерированные word2vec относительно переведенных названий сущностей с настройками MultiKE. Можно видеть, что появилось гораздо больше пересечений синих и красных пятен, что говорит о лучшем качестве сопоставления названий сущностей. Однако результат метода генерации векторных представлений MultiKE имеет выраженные пятна одного цвета, что говорит о невысокой точности этого метода. Аналогичные результаты можно наблюдать на примере метода RDGCN.

На рис. 4 показаны векторные представления для имен сущностей из набора данных EN_RU_15K_V1, сгенерированные при помощи word2vec с оригинальными настройками RDGCN (без перевода названий сущностей).

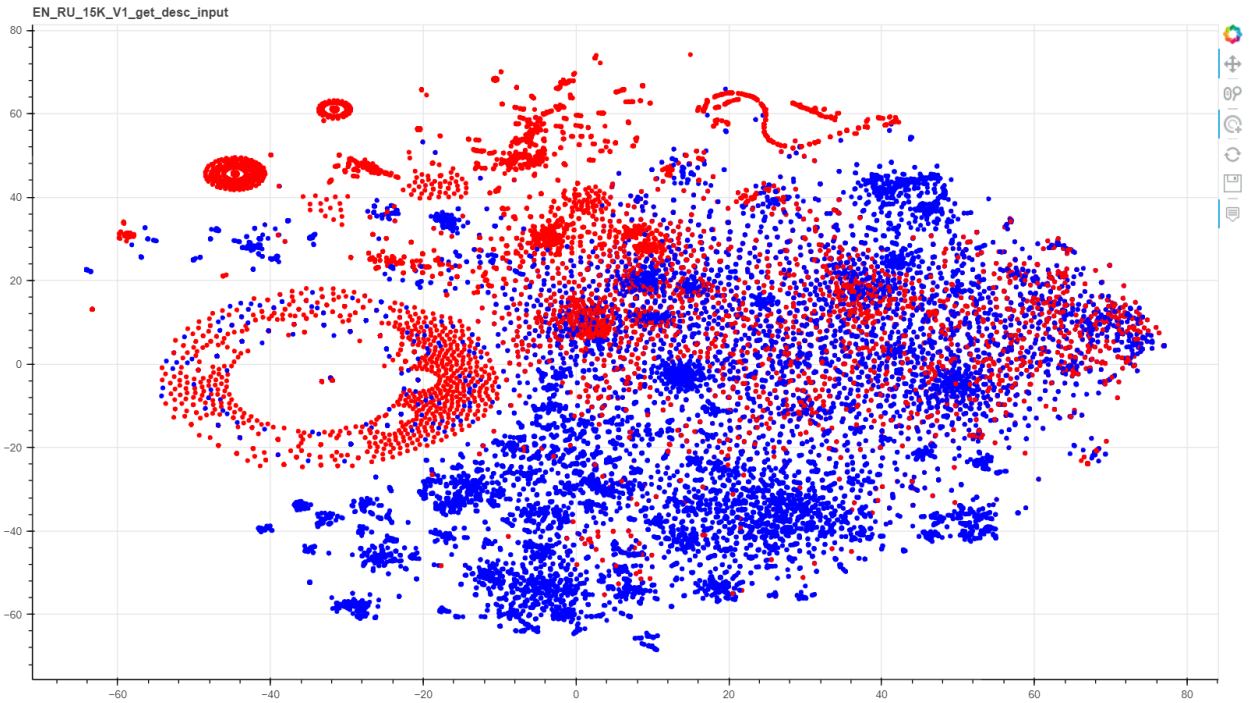


Рис. 4. Эмбеддинги имен сущностей, из набора данных EN_RU_15K_V1, построенные генератором векторных представлений имен сущностей RDGCN с оригинальными настройками (без перевода).

На Рис. 5 показаны векторные представления имен сущностей из этого же набора данных, сгенерированные word2vec относительно переведенных названий сущностей с настройками RGDCN. На этом рисунке в левом нижнем углу имеется кластер эллипсоидной формы. Он возник из-за зануления векторов слов, для

которых алгоритм из RGDCN не нашел значений в предобученной модели. В остальном же данное векторное представление имеет большую степень наложения по сравнению с RGDCN.

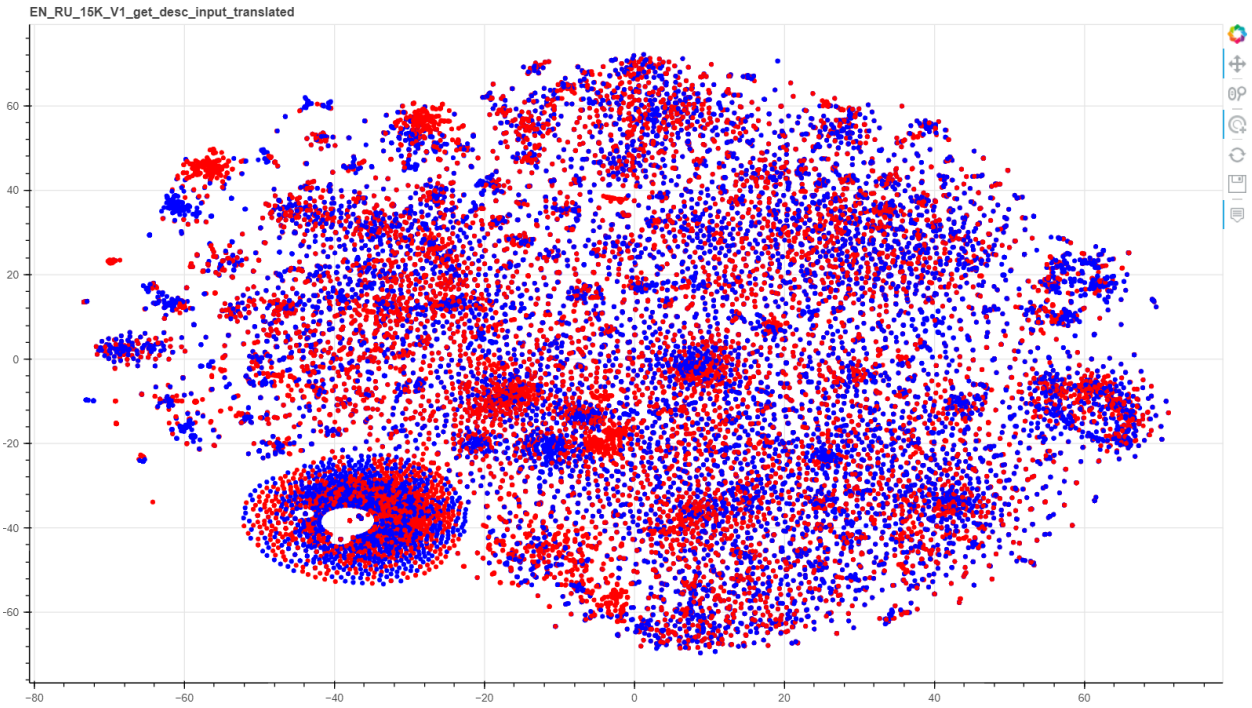


Рис. 5. Эмбеддинги имен сущностей, из набора данных EN_RU_15K_V1, построенные генератором векторных представлений имен сущностей RDGCN с оригинальными настройками и с переводом.

Наконец, на рис. 6 показаны векторные представления имен сущностей из набора данных EN_RU_15K_V1, построенные генератором векторных представлений имен сущностей SEU с оригинальными настройками и переводом. Легко видеть, что это представление имен сущностей дает наилучшее соответствие между англоязычными и русскоязычными именами сущностей. Красные пятна возникают только в случае существенного различия в названиях сущностей. Например, часть сущностей типа *Film* имеет абсолютно другое русскоязычное название. Для сравнения:

- Англоязычное название «The_Death_and_Life_of_Bobby_Z» и русскоязычное «Подстава_(фильм,_2007)»;
- Англоязычное название «The Break-Up» и русскоязычное «Развод по-американски(фильм,_2006)»;

- Англоязычное название «The_Beyond_(film)» и русскоязычное «Седьмые_врата_ада».

Понятно, что переводчик не может должным образом учитывать такие ситуации.

Аналогичная ситуация возникает с футбольными клубами. Здесь также к ошибкам встраивания могут приводить сокращения “FC@ и тег «футбольный_клуб». Например, в пространстве эмбеддингов достточно далеко расположены:

- Англоязычное название «Olympique_Club_de_Khouribga» и русскоязычное «Хурибга_(футбольный_клуб)»;
- Англоязычное название «FC_Tosno» и русскоязычное «Тосно_(футбольный_клуб)» ;
- Англоязычное название «FC_Balzers» и русскоязычное «Бальцерс_(футбольный_клуб)».

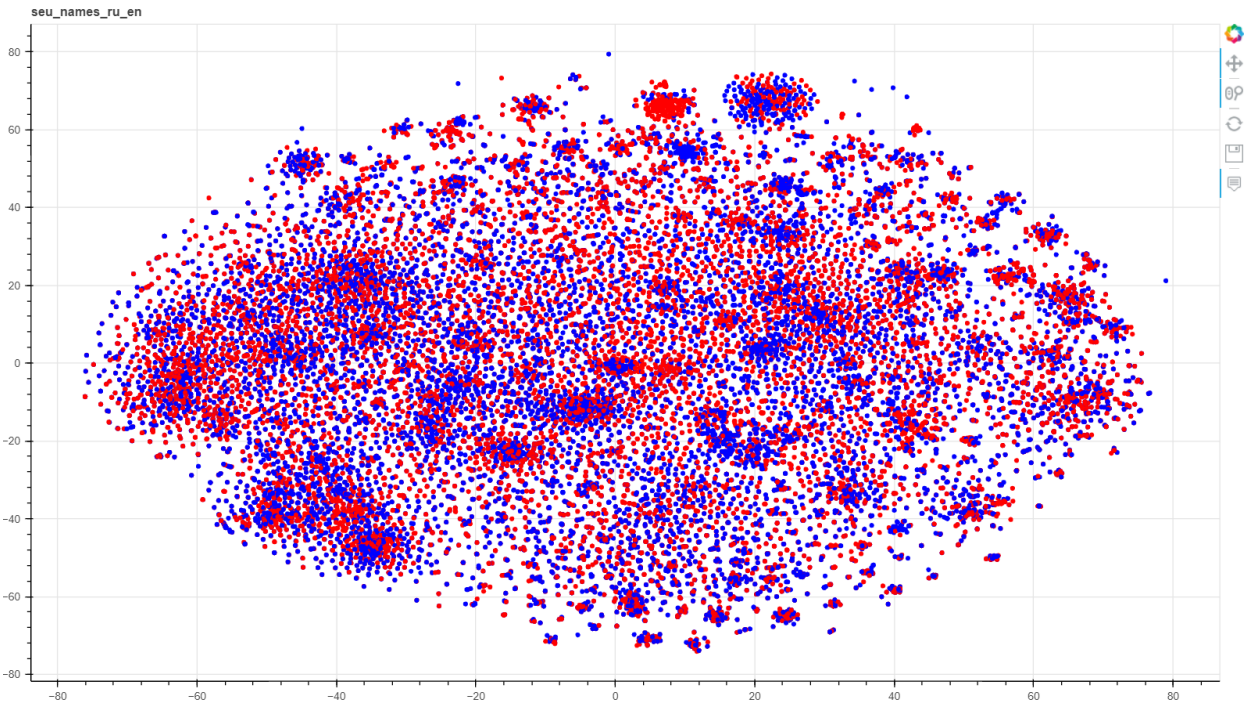


Рис. 6. Эмбеддинги имен сущностей из набора данных EN_RU_15K_V1, построенные генератором векторных представлений имен сущностей SEU с оригинальными настройками и с переводом.

Более подробно качество результатов выравнивания сущностей в зависимости от наличия или отсутствия перевода названий сущностей показано в Таблице 1. В столбце Перевод (Пер.) знаком плюс или минус обозначен факт наличия или отсутствия перевода имен сущностей.

Таблица 1. Влияние перевода имен сущностей на качество алгоритмов EA

Метод	Набор данных	Пер.	hits@1	hits@10	hits@50	mrr	Улучш.
multiKE	EN_FR_15K_V1	-	74,133	83,59	88,867	0,774435	
multiKE	EN_FR_15K_V1	+	74,619	84,324	89,638	0,779689	0,486
multiKE	EN_FR_15K_V2	-	85,495	92,057	95,276	0,878035	
MultiKE	EN_FR_15K_V2	+	85,952	92,238	95,505	0,882119	0,457
MultiKE	EN_RU_15K_V1	-	31,544	45,711	59,933	0,364153	
MultiKE	EN_RU_15K_V1	+	35,667	51,111	63,856	0,409273	4,123
MultiKE	EN_RU_15K_V2	-	45,3	62,289	74,244	0,510486	
MultiKE	EN_RU_15K_V2	+	46,478	62,289	73,956	0,519488	1,178
multiKE	EN_RU_100K_V1	-	16,262	24,038	33,145	0,190415	
Multi	EN_RU_100K_V1	+	19,568	30,158	41,72	0,232864	3,306
Rdgcn	EN_FR_15K_V1	-	77,019	89,181	92,438	0,813097	
Rdgcn	EN_FR_15K_V1	+	76,905	89,324	92,448	0,813125	-0,114
Rdgcn	EN_FR_15K_V2	-	86,19	94,848	97,124	0,895109	
Rdgcn	EN_FR_15K_V2	+	87,095	95,114	97,257	0,902504	0,905
Rdgcn	EN_RU_15K_V1	-	39,633	59,667	71,2	0,460294	
Rdgcn	EN_RU_15K_V1	+	74,378	88,211	92,322	0,791784	34,745
Rdgcn	EN_RU_15K_V2	-	53,656	71,689	80,322	0,599311	
Rdgcn	EN_RU_15K_V2	+	84,378	92,3	96,667	0,88172	30,722

Данные эксперименты показали, что перевод названий сущностей имеет существенное влияние на качество алгоритмов выравнивания сущностей, но имеются и другие параметры, влияющие на качество этих алгоритмов. Возник вопрос, влияют ли на качество методов выравнивания сущностей сами методы генерации эмбеддингов имен сущностей?

Поэтому в дальнейшем были поробно рассмотрены различные способы построения векторных представлений имен сущностей, а также комбинации различных стратегий построения векторных представлений на основе реляционных триплет с различными вариантами построения векторных представлений для имен сущностей.

4. КАЧЕСТВО МЕТОДОВ ВЫРАВНИВАНИЯ СУЩНОСТЕЙ В ЗАВИСИМОСТИ ОТ РАЗНЫХ СПОСОБОВ ПОСТРОЕНИЯ ВЕКТОРНЫХ ПРЕДСТАВЛЕНИЙ ИМЕН СУЩНОСТЕЙ

Было рассмотрено несколько моделей генерации векторных представлений имен сущностей, которые можно использовать для целей выравнивания сущностей. Было выделено несколько типов генераторов векторных представлений имен сущностей.

Генератор 1. Генерация векторных представлений уровня слов (модель word2vec) и уровня литер (модель fastText). Этот генератор применяется в методе выравнивания сущностей multiKE.

Генератор 2. Перевод имен сущностей на английский язык, генерация векторных представлений на уровне слов (модель glove.840B.300d). Этот генератор применяется в методе выравнивания сущностей RDGCN.

Генератор 3. За основу берется предположение, что не только информация о структуре окрестностей, но и текстовая информация эквивалентных сущностей обладают свойством изоморфизма. Построение векторного представления имен сущностей состоит из следующих этапов: перевод входных данных на английский язык, чтение предобученной модели предобработка входных данных, токенизация по словам, формирование биграмм, формирование векторных представлений, снижение размерности. В качестве предобученной модели использовалась glove.6B.300d. Генератор применяется в методе выравнивания SEU.

Также в качестве альтернативных методов генерации векторных представлений имен сущностей были выбраны современные модели обработки естественных языков XLNet [17] и LaBSE [18]. Спецификой этих моделей является возможность строить векторные представления для наборов слов, таких как предложения.

Генератор 4. (XLNet). Целью модели XLNet является изучение распределений для всех перестановок слов в заданной последовательности. Векторные представления формируются в рамках только одного языка, поэтому для решения нашей задачи потребовалось предварительно применить машинный перевод.

Генератор 5. (LaBSE). Данная модель генерирует независимые от языка векторные представления предложений на основе модели BERT. Представление создается путём объединения возможностей маскированного и кросс-языкового моделирований [9].

Эти пять генераторов использовались для генерации векторных представлений имен сущностей, а затем полученные представления имен сущностей встраивались в три различных алгоритма выравнивания сущностей, а именно, MultiKE, RDGCN и SEU. Для оценки качества полученных резхультатов вычислялись метрики hits@k и MRR, а также строились визуализации результатов. В таблице 2 показаны оценки качества работы трех алгоритмов выравнивания сущностей в зависимости от используемого генератора векторных представлений имен сущностей. Таблица 2 демонстрирует, что генератор векторных представлений имен сущностей на основе модели LaBSE показал себя достаточно хорошо. Он оказался эффективнее генераторов 1 и 2.

Таблица 2. Оценки качества работы алгоритмов выравнивания сущностей в зависимости от используемого генератора векторных представлений имен сущностей

Метод	Ген.	Hits@1	Hits@5	Hits@10	Hits@50	MRR
MultiKE	1	52.0	62.1	66.6	76.9	0,570
MultiKE	2	69.9	78.1	81.3	87.8	0,737
MultiKE	3	81.2	87.5	89.1	93.2	0,841
RDGCN	2	74.4	84.7	88.2	92.3	0,792
RDGCN	1	68.0	79.6	82.8	88.4	0,733
RDGCN	3	84.8	92.1	93.5	95.6	0,881
RDGCN	4	43.4	50.0	53.0	60.5	0,467
RDGCN	5	75.4	83.7	85.9	89.7	0,792
SEU	3	97.2	99.1	99.5	99.8	0,981
SEU	1	88.1	93.5	94.8	97.5	0,905
SEU	2	87.4	93.1	95.4	98.6	0,905
SEU	4	32.5	41.3	45.5	54.9	0,369
SEU	5	094.9	97.6	98.4	99.3	0,962

Тем не менее, генератор 3 оказался наиболее эффективным. Методы выравнивания сущностей MultiKE и RDGCN на его основе превысили исходные значения точности. Модель XLNet оказалась непригодной для целей выравнивания сущностей, так как полученные с ее помощью оценки точности методов выравнивания сущностей почти в два раза хуже остальных методов. Результаты подходов на ее основе близки к значениям, полученным без перевода. Эти выводы подтверждаются и визуализациями результатов методов выравнивания сущностей, использующих разные языковые модели. Для сравнения на рис. 7 показана визуализация результата работы метода выравнивания RDGCN с использованием генератора 5 (языковая модель XLNet), который соответствует наихудшим оценкам качества выравнивания. На рис. 8 показана визуализация результата работы метода выравнивания RDGCN с использованием генератора 3 (языковая модель, используемая методом SEU), который соответствует наилучшим оценкам качества.

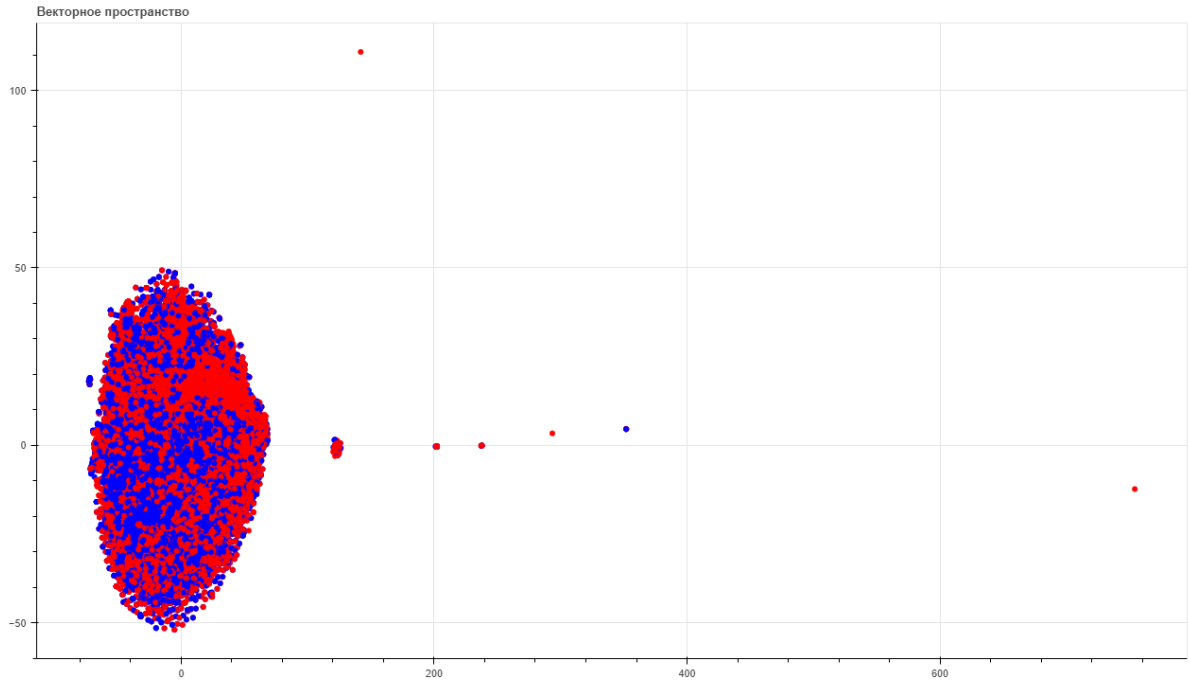


Рис. 7. Визуализация результата работы метода выравнивания RDGCN с использованием генератора 5 (языковая модель XLNet), который соответствует наихудшим оценкам качества.

Результаты применения моделей XLNet и LaBSE к MultiKE не указаны в связи с нехваткой вычислительных ресурсов для построения векторных представлений

литералов. Выводы об их эффективности сделаны на основе значений, полученных на базе других подходов.

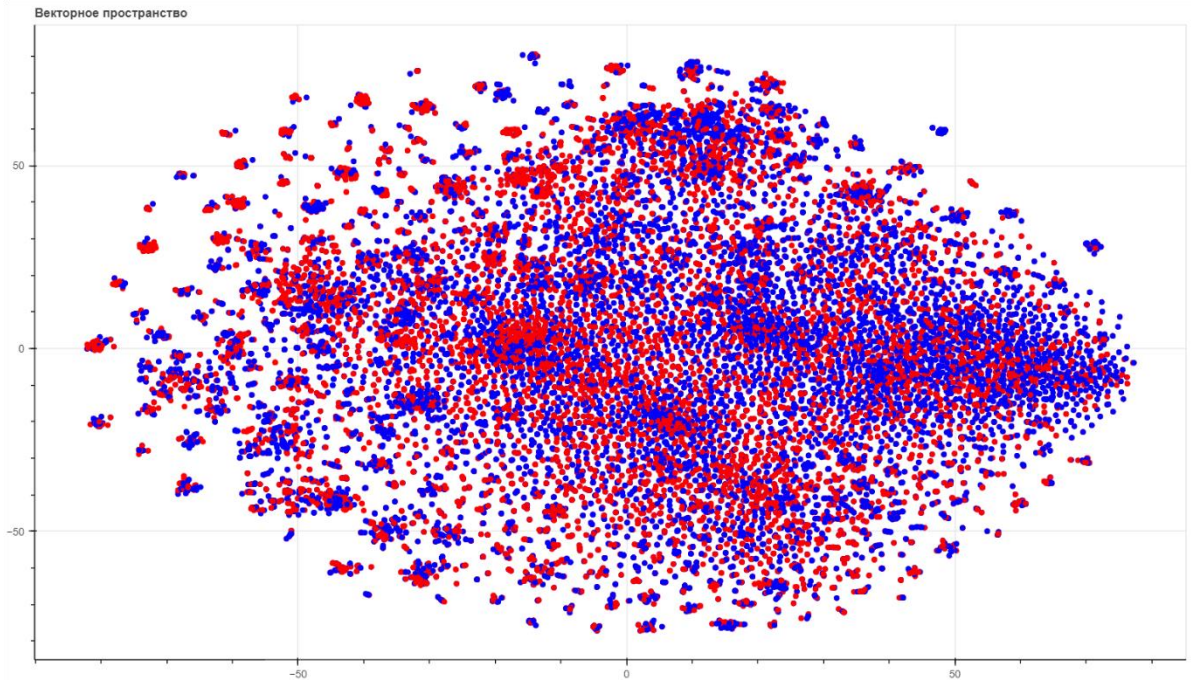


Рис. 8. Визуализация результата работы метода выравнивания RDGCN с использованием генератора 3 (языковая модель, используемая методом выравнивания SEU), который соответствует наилучшим оценкам качества выравнивания.

5. КАЧЕСТВО АЛГОРИТМОВ ВЫРАВНИВАНИЯ СУЩНОСТЕЙ В ЗАВИСИМОСТИ ОТ ТИПОВ СУЩНОСТЕЙ И КОЛИЧЕСТВА ОТНОШЕНИЙ МЕЖДУ СУЩНОСТЯМИ

Для более детального анализа качества алгоритмов выравнивания сущностей была произведена оценка метрик качества по отдельным типам сущностей, количеству отношений и атрибутов.

Для определения типов сущностей использовались файлы «instance_types». Следует отметить, что в разных языковых версиях данных DBpedia часто наблюдаются несоответствия между типами сущностей, связанных отношением *owl:sameAs*. Например, русскоязычная сущность «Эминем» отнесена к типу «MusicalArtist», а ее английский эквивалент «Eminem» относится к вышестоящему по иерархии типу «Person». Было установлено, что только шестьдесят семь процентов эквивалентных сущностей отнесены к одному и тому же типу.

Для решения указанной проблемы была написана программа установления общих типов для сущностей, связанных отношением *owl:sameAs*. Для этого при помощи SPARQL-запроса к англоязычной DBpedia была построена иерархия типов DBpedia. Для каждой пары эквивалентных сущностей осуществлялось сравнение приписанных им типов. В случае, когда ни одна сущность из пары не являлась подтипом другой, но при этом у них имелся общий надтип, им обеим приписывался последний. Например, сущность «Воеводина» относится к типу «AdministrativeRegion», а ее английский эквивалент «Vojvodina» относится к типу «Country». Данным сущностям присвоится общий тип «PopulatedPlace». В результате указанной процедуры в наборе данных EN-RU-15K (V1) было выделено семьдесят три типа сущностей.

На рисунках 8 и 9 приведены значения метрики Hits@1, показывающие качество выравнивания сущностей разного типа, полученные при помощи методов MultiKE и RDGCN. В обоих случаях использовался Генератор 3 векторных представлений имен сущностей. Информация представлена для типов, насчитывающих больше 100 сущностей.

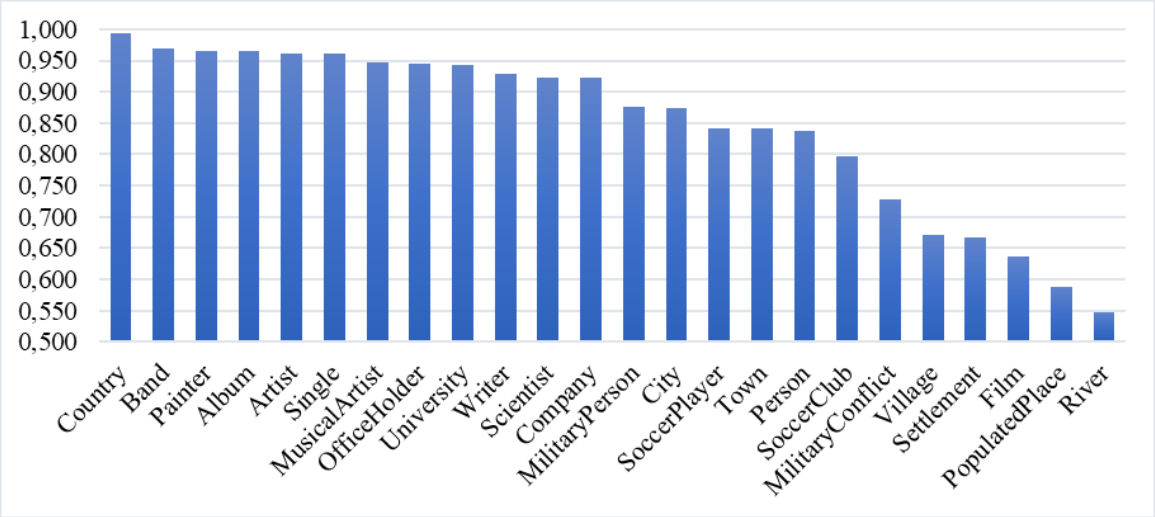


Рис. 9. Значения Hits@1 для разных типов сущностей, полученные методом MultiKE с Генератором 3.

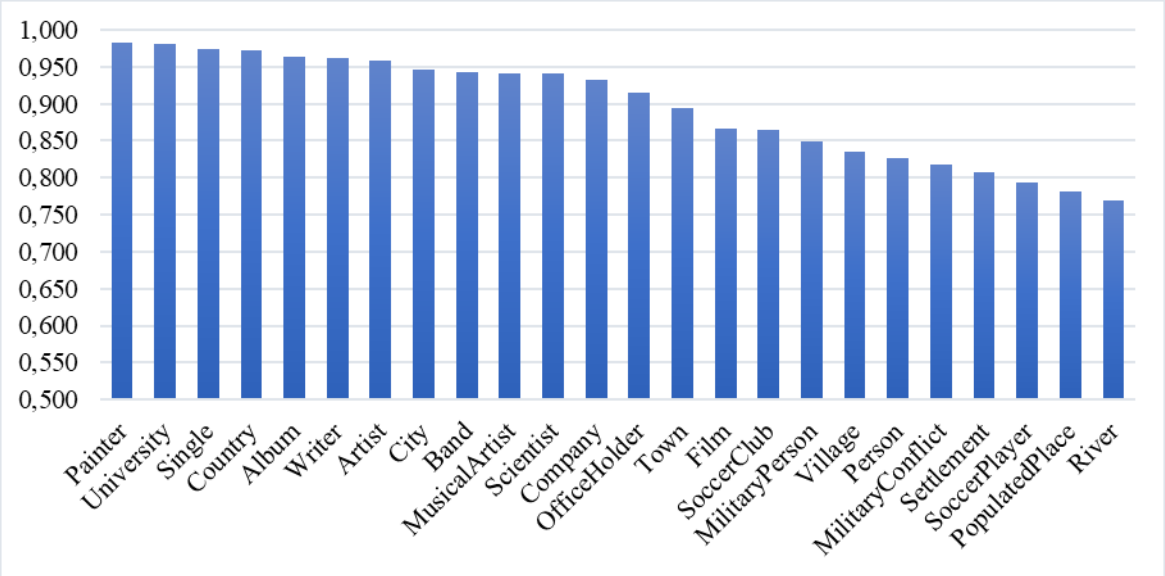


Рис. 10. Значения Hits@1 для разных типов сущностей, полученные методом RDGCN с Генератором 3.

Можно видеть, что типы, дающие наилучшие оценки точности выравнивания, похожи для обоих методов выравнивания, как и типы, дающие наихудшие оценки. При этом более равномерное распределение точности в зависимости от типов сущностей демонстрирует метод RDGCN.

Тем не менее, наилучшую точность методов выравнивания по отдельным типам показал метод SEU. Из семидесяти трех типов сорок один тип выдал оценку 100% для hits@1. Значения hits@1, hits@10 и среднеобратного ранга (MRR) для остальных типов сущностей показаны в Таблице 3.

Таблица 3. Значения hits@1, hits@10 и среднеобратного ранга (MRR) для типов сущностей, полученные методом SEU с Генератором 3.

type	hits1	hits10	Mrr
AdultActor	0	0	4,761905
Award	0	100	33,33333
Insect	44,44444	77,77778	57,48534
Mammal	60	100	76,66667
Bird	66,66667	100	80,55556
GovernmentAgency	75	100	81,25
Reptile	78,57143	100	86,90476

Noble	83,33333	100	87,5
River	86,31579	96,84211	90,60496
MusicalWork	90,90909	95,45455	92,3951
Town	92,0354	99,11504	94,69713
MilitaryConflict	94,0678	99,57627	96,13459
PopulatedPlace	94,45471	99,07579	96,18643
AdministrativeRegion	94,73684	98,68421	95,89808
Film	94,8728	98,98239	96,35096
City	94,97908	99,16318	96,99585
Writer	95,3125	99,21875	97,05116
Village	95,6044	100	97,61905
Settlement	95,78488	99,49128	97,01837
SoccerClub	96,90141	99,43662	97,75137
Royalty	97,4359	100	98,2906
Scientist	97,64706	100	98,82353
SoccerPlayer	97,66472	99,83319	98,46351
Person	97,73196	99,38144	98,37685
MilitaryPerson	98,3871	98,92473	98,72632
Band	98,64603	99,41973	98,90352
MusicalArtist	98,73817	99,68454	99,16708
Company	99,03846	99,51923	99,10003
Album	99,21569	99,66387	99,39928
OfficeHolder	99,28401	99,76134	99,43016
Artist	99,65724	100	99,80291
Single	99,7815	100	99,8689

Как и ранее, более подробный анализ показал, что наихудшие значения выравнивания возникали на сущностях с низким соответствием по названиям.

6. ЗАКЛЮЧЕНИЕ

Мы исследовали влияние методов построения векторных представлений (эмбеддингов) для имён сущностей и литералов на качество результатов различных методов выравнивания сущностей. Был исследован вклад применения перевода и современных моделей обработки естественных языков, таких как LabSE и XLnet. Для интуитивного понимания результатов было построено значительное количество визуализаций. Также эксперименты показали, что количество отношений и атрибутов сущностей в разных наборах данных не влияет на качество выравнивания. Скорее, имеет значение количество совпадающих отношений и атрибутов. В настоящее время разрабатывается новый инструмент визуализации, который позволит анализировать отношения отдельных сущностей и их влияние на качество алгоритмов выравнивания.

СПИСОК ЛИТЕРАТУРЫ

1. *Sun Z., Zhang Q., Hu W., Wang C., Chen M., Akrami F. et al.* A benchmarking study of embedding-based entity alignment for knowledge graphs // Proc. VLDB Endowment. 2020. Vol. 13. P. 2326–2340.

2. *Gnezdilova V.A., Apanovich Z.V.,* Russian-English dataset and comparative analysis of algorithms for cross-language embedding-based entity alignment // Journal of Physics: Conference Series. 2021. Vol. 2099.

3. *Zhang Q., Sun Z., Hu W., Chen M., Guo L. et al.* Multi-view knowledge graph embedding for entity alignment // Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. 2019. P. 5429–5435.

4. *Mikolov T., Chen K., Corrado G., Dean J.* Efficient estimation of word representations in vector space, January 2013, URL: https://arxiv.org/abs/1301.3781.

5. *Bordes A., Usunier N., Garcia-Durán A, Weston J., Yakhnenko O.* Translating embeddings for modeling multi-relational data // Proceedings of the 26th International Conference on Neural Information Processing Systems. 2013. Vol. 2. P. 2787–2795.

6. *Wu Y., Liu X., Feng Y., Wang Z., Yan R., Zhao D.* Relation-aware entity alignment for heterogeneous knowledge graphs // Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. 2019. P. 5278–5284.

7. *Veličković P., Cucurull G., Casanova A., Romero A., Liò P., Bengio Y.* Graph attention networks// ICLR. 2018. 12 p.

8. *Wang Z., Lv Q., Lan X., Zhang Y.* Cross-lingual knowledge graph alignment via graph convolutional networks // Proc. of the Conference on Empirical Methods in Natural Language Processing. 201., P. 349–357.

9. *Mao X., Wang W., Wu Y., Lan M.* From alignment to assignment: frustratingly simple unsupervised entity alignment // Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021. P. 2843–2853.

10. *Xu K., Wang L., Yu M., Feng Y., Song Y., et al.* Cross-lingual knowledge graph alignment via graph matching neural network // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. P. 3156–3161.

11. *Pennington J, Socher R., Manning C.D.* GloVe: Global Vectors for Word Representation // Conference on Empirical Methods in Natural Language. 2014. P. 1532-1543.

12. *Bojanowski P., Grave E., Joulin A., Mikolov T.* Enriching word vectors with subword information // Transactions of the Association for Computational Linguistics. 2017. P. 135–146.

13. *Devlin J., Chang M.-W., Lee K., Toutanova K.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2019. Vol. 1. P. 4171–4186.

14. *Fuglede B., Topsoe F.* Jensen–Shannon divergence and Hilbert space embedding // Proceedings of the International Symposium on Information Theory, 2004. IEEE.

15. *Sun Z., Hu W., Zhang Q., Qu Y.* Bootstrapping entity alignment with knowledge graph embedding // Proc. 27th International Joint Conference on Artificial Intelligence (IJCAI-18), P. 4396–4402.

16. *Guo L., Sun Z., Hu W.* Learning to Exploit Long-term relational dependencies in knowledge graphs // Proceedings of the 36th International Conference on Machine Learning. 2019. Vol. 57. P. 2505–2514.

17. *Maaten L. van der, Hinton G.* Visualizing data using t-SNE // Journal of Machine LearningResearch. 2008. Vol. 86. P. 2579–2605.

18. *Yang Z., Dai Z., Yang Y., Carbonell J., Salakhutdinov R. et al.* XLNet: generalized autoregressive pretraining for language understanding // Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019. P. 5753–5763.

19. *Feng F., Yang Y., Cer D., Arivazhagan N0, Wang W.* Language-agnostic BERT sentence embedding. 2020. URL: https://arxiv.org/abs/2007.01852.

HOW ENTITY NAME EMBEDINGS AFFECT THE QUALITY OF ENTITY ALIGNMENT

D. I. Gusev[1] [0000-0001-9636-2783], Z. V. Apanovich[2] [0000-0002-5767-284X]

[1]Novosibirsk State University, Novosibirsk;
[2]A.P. Ershov Institute of Informatics Systems, Siberian Branch of the Russian Academy of Sciences, Novosibirsk State University, Novosibirsk

[1]d.gusev1@g.nsu.ru, [2]apanovich_09@mail.ru

Abstract

Cross-lingual entity alignment algorithms are designed to look for identical real-world objects in multilingual knowledge graphs. This problem occurs, for example, when searching for drugs manufactured in different countries under different names, or when searching for imported equipment. At the moment, there are several open-source libraries that collect implementations of entity alignment algorithms as well as test data sets for various languages. This paper describes experiments with several popular entity alignment algorithms applied to a Russian-English dataset. In addition to translating entity names from Russian to English, experiments on combining the various generators of entity name embeddings with the various generators of relational information embeddings have been conducted. In order to obtain more detailed information about the results of the EA approaches, an assessment by entity types, the number of relationships and attributes have been made. These experiments allowed us to significantly improve the accuracy of several EA algorithms on the English-Russian dataset.

Keywords multi-lingual knowledge graphs, identity resolution, cross-lingual entity alignment, relational embeddings, name embeddings correctness

REFERENCES

1. *Sun Z., Zhang Q., Hu W., Wang C., Chen M., Akrami F. et al.* A benchmarking study of embedding-based entity alignment for knowledge graphs // Proc. VLDB Endowment. 2020. Vol. 13. P. 2326–2340.

2. *Gnezdilova V.A., Apanovich Z.V.,* Russian-English dataset and comparative analysis of algorithms for cross-language embedding-based entity alignment // Journal of Physics: Conference Series. 2021. Vol. 2099.

3. *Zhang Q., Sun Z., Hu W., Chen M., Guo L. et al.* Multi-view knowledge graph embedding for entity alignment // Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. 2019. P. 5429–5435.

4. *Mikolov T., Chen K., Corrado G., Dean J.* Efficient estimation of word representations in vector space, January 2013, URL: https://arxiv.org/abs/1301.3781.

5. *Bordes A., Usunier N., Garcia-Durán A, Weston J., Yakhnenko O.* Translating embeddings for modeling multi-relational data // Proceedings of the 26th International Conference on Neural Information Processing Systems. 2013. Vol. 2. P. 2787–2795.

6. *Wu Y., Liu X., Feng Y., Wang Z., Yan R., Zhao D.* Relation-aware entity alignment for heterogeneous knowledge graphs // Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. 2019. P. 5278–5284.

7. *Veličković P., Cucurull G., Casanova A., Romero A., Liò P., Bengio Y.* Graph attention networks// ICLR. 2018. 12 p.

8. *Wang Z., Lv Q., Lan X., Zhang Y.* Cross-lingual knowledge graph alignment via graph convolutional networks // Proc. of the Conference on Empirical Methods in Natural Language Processing. 201., P. 349–357.

9. *Mao X., Wang W., Wu Y., Lan M.* From alignment to assignment: frustratingly simple unsupervised entity alignment // Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021. P. 2843–2853.

10. *Xu K., Wang L., Yu M., Feng Y., Song Y., et al.* Cross-lingual knowledge graph alignment via graph matching neural network // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. P. 3156–3161.

11. *Pennington J, Socher R., Manning C.D.* GloVe: Global Vectors for Word Representation // Conference on Empirical Methods in Natural Language. 2014. P. 1532-1543.

12. *Bojanowski P., Grave E., Joulin A., Mikolov T.* Enriching word vectors with subword information // Transactions of the Association for Computational Linguistics. 2017. P. 135–146.

13. *Devlin J., Chang M.-W., Lee K., Toutanova K.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2019. Vol. 1. P. 4171–4186.

14. *Fuglede B., Topsoe F.* Jensen–Shannon divergence and Hilbert space embedding // Proceedings of the International Symposium on Information Theory, 2004. IEEE.

15. *Sun Z., Hu W., Zhang Q., Qu Y.* Bootstrapping entity alignment with knowledge graph embedding // Proc. 27th International Joint Conference on Artificial Intelligence (IJCAI-18), P. 4396–4402.

16. *Guo L., Sun Z., Hu W.* Learning to Exploit Long-term relational dependencies in knowledge graphs // Proceedings of the 36th International Conference on Machine Learning. 2019. Vol. 57. P. 2505–2514.

17. *Maaten L. van der, Hinton G.* Visualizing data using t-SNE // Journal of Machine LearningResearch. 2008. Vol. 86. P. 2579–2605.

18. *Yang Z., Dai Z., Yang Y., Carbonell J., Salakhutdinov R. et al.* XLNet: generalized autoregressive pretraining for language understanding // Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019. P. 5753–5763.

19. *Feng F., Yang Y., Cer D., Arivazhagan N0, Wang W.* Language-agnostic BERT sentence embedding. 2020. URL: https://arxiv.org/abs/2007.01852.

СВЕДЕНИЯ ОБ АВТОРАХ

ГУСЕВ Даниил Иванович – магистрант Новосибирского государственного университета. Сфера научных интересов – визуализация информации, Semantic Web.

Daniil Ivanovic GUSEV – Masters student of Novosibirsk State University. Research interests include information visualization, Semantic Web.

email: d.gusev1@g.nsu.ru

ORCID: 0000-0001-9636-2783

АПАНОВИЧ Зинаида Владимировна – старший научный сотрудник Института Систем Информатики СО РАН, доцент Новосибирского государственного университета. Сфера научных интересов – визуализация информации, визуализация графов, Semantic Web.

Zinaida Vladimirovna APANOVICH – senior researcher of the Institute of Informatics Systems of SB RAS, Associate Professor of Novosibirsk State University. Research interests include information visualization, graph visualization, Semantic Web.

email: apanovich@iis.nsk.su

ORCID: 0000-0002-5767-284X

Материал поступил в редакцию 12 января 2023 года

УДК 013, 004.65

УНИФИЦИРОВАННОЕ ПРЕДСТАВЛЕНИЕ ОНТОЛОГИИ ЕДИНОГО ЦИФРОВОГО ПРОСТРАНСТВА НАУЧНЫХ ЗНАНИЙ

Н. Е. Каленов[1] [0000-0001-5269-0988], А. Н. Сотников[2] [0000-0002-0137-1255]

[1, 2]*Межведомственный суперкомпьютерный центр (МСЦ) РАН – филиал ФГУ ФНЦ Научно-исследовательский институт системных исследований (НИИСИ) РАН, Ленинский пр., 32а, г. Москва, 119334*

[1]nkalenov@jscc.ru, [2]asotnikov@jscc.ru

Аннотация

Единое цифровое пространство научных знаний (ЕЦПНЗ) представляет собой цифровую информационную среду, агрегирующую разнородную информацию, связанную с различными аспектами научных знаний. Одной из важных функций ЕЦПНЗ является предоставление информации для решения задач искусственного интеллекта, что обусловливает необходимость поддержки данных в структуре, соответствующей правилам Semantic Web. Особенностями ЕЦПНЗ являются, с одной стороны, политематичность и разнородность элементов контента, с другой – высокая динамика появления новых видов объектов и связей между ними, что обусловлено спецификой развития науки. При реализации ЕЦПНЗ должна быть обеспечена возможность навигации по разнородным ресурсам пространства с использованием семантических связей между ними. Возможности ЕЦПНЗ в значительной мере определяются структурой онтологии пространства, модель которой предложена в данной работе. В рамках модели проведена иерархическая структуризация онтологии ЕЦПНЗ; выделены и определены такие элементы, как «подпространство», «класс объектов», «объект», «атрибуты объекта», три типа попарных связей объектов и атрибутов (универсальные, квазиуниверсальные и специфические). Структура каждого типа элементов определяется «справочником» унифицированного вида; конкретные значения атрибутов и связей содержатся в словарях унифицированной структуры. Выделен класс объектов «Форматы», описывающих правила формирования атрибутов и значений связей. Пред-

ложена формализация представлений справочников и словарей ЕЦПНЗ. Предлагаемая модель позволяет достаточно просто добавлять в пространство, по мере необходимости, новые виды объектов, их попарных связей и атрибутов.

***Ключевые слова**: цифровое пространство научных знаний, онтологии, структуризация, связанные данные, атрибуты данных, семантический WEB.*

ВВЕДЕНИЕ

Единое цифровое пространство научных знаний (ЕЦПНЗ) представляет собой цифровую среду, агрегирующую разнородную информацию, связанную с различными аспектами научных знаний. ЕЦПНЗ должно обеспечить поддержку процессов предоставления широкому кругу пользователей необходимой им информации в различных областях науки. ЕЦПНЗ рассматривается как интегратор для научных целей государственных информационных систем (ИС) (таких как Большая Российская энциклопедия, Национальная электронная библиотека, Государственный каталог географических названий и пр.) с отраслевыми научными информационными ИС, электронными библиотеками (ЭБ), регистрами и т. п. В рамках ЕЦПНЗ необходимо объединить эти ресурсы на основе онтологического подхода и Semantic Web для решения широкого круга образовательных и научных задач, в том числе, ориентированных на применение методов искусственного интеллекта.

Отличительной особенностью ЕЦПНЗ являются политематичность и разнородность элементов контента с обеспечением возможности навигации по ресурсам пространства с использованием семантических связей между ними.

Программная оболочка ЕЦПНЗ должна обрабатывать широкий спектр запросов, не обязательно содержащих термины, в явном виде присутствующие в метаданных, относящихся к конкретным объектам ЕЦПНЗ. Например, на запрос «археологические находки в Западной Сибири в 20 веке» должны быть выданы описания всех археологических объектов, найденных в Томской, Новосибирской областях, в Тобольске и т. д., за период с 1901 по 2000 годы. При этом в информации об отдельном объекте может содержаться указание лишь на конкретное место его обнаружения, а заключение о том, что данное место относится к Западной Сибири, вытекает из автоматического анализа связей между объектами пространства (в данном случае относящимися к географии и времени).

Цели создания, задачи и общие принципы построения ЕЦПНЗ приведены в [1–3].

Одним из первых шагов к практической реализации ЕЦПНЗ является разработка его онтологии – определение правил формирования его составляющих, включая наполнение контента разнородными, но связанными по единым правилам, данными. Общие подходы к формированию онтологии ЕЦПНЗ отражены в [4, 5].

Построению онтологий и правилам их отражения в Сети посвящено значительное количество исследований и публикаций. В рамках Simple Knowledge Organization System (SKOS) [6–8] разработаны формальные правила отражения в цифровой среде связанных открытых данных (LOD), тезаурусов, свойств объектов и их связей с использованием правил OWL и RDF. Примеры многочисленных реализаций онтологического подхода, разработанного в рамках SKOS применительно к различным областям человеческой деятельности (пищевая промышленность, музейное дело, география, социальные науки и т. д.), отражены в [9–13].

На сайте «онтологического форума» [14] ежедневно появляется информация о семинарах, симпозиумах, рабочих встречах и т. п., посвященных проблемам создания онтологий в различных сферах человеческой деятельности.

Хотя многочисленные реализации онтологий, представленные в интернете, построены по общим принципам, каждая из них строится независимо. И обеспечить на практике интеграцию ресурсов, построенных на основе этих онтологий, достаточно затруднительно. Примеров такой интеграции нам обнаружить не удалось.

ЕЦПНЗ, в отличие от других информационных систем, должно обеспечивать реальную интеграцию разнородных данных. Этого можно достичь, только используя унифицированную, четкую и в то же время достаточно простую технологию формирования онтологии пространства в целом и его отдельных составляющих. Вариант такой технологии, не противоречащей принципиальным подходам SKOS и OWL, но являющейся фактически их развитием в сторону упрощения модели, предложен ниже.

1. Общие понятия

ЕЦПНЗ рассматривается как иерархическая структура, включающая подпространства, классы объектов, объекты, атрибуты объектов, значения атрибутов объектов. Наряду с этой структурой имеется структура попарных связей объектов и попарных связей значений атрибутов. Каждая связь, в свою очередь, имеет свое значение и может иметь атрибуты и их значения.

Все перечисленные составляющие назовем элементами ЕЦПНЗ. Каждый элемент имеет своё уникальное имя (URN).

Подпространство – это совокупность элементов ЕЦПНЗ, относящихся к определенному научному направлению; выделяется универсальное подпространство, содержащее информацию об объектах мультидисциплинарного характера (персоны, события, единицы измерения и т. п.).

Тематическое подпространство (например, подпространство «информатика», «космические исследования», «химия» и др.) содержит элементы, напрямую связанные с данным научным направлением, а также связи с элементами универсального и других тематических подпространств, и включает политематические и общенаучные объекты.

Объект – совокупность структурированной многоаспектной информации о физической сущности (например, о конкретном человеке, конкретной книге, музейном предмете и т. п.), научном понятии (например, об уравнении Матье, Законе всемирного тяготения, корпусе текстов китайского языка и т. п.), событии или научном мероприятии и др. Объект как понятие может рассматриваться как аналог энциклопедического «слота», который также может являться объектом ЕЦПНЗ. Каждый объект характеризуется своими значениями атрибутов и связей с другими объектами.

Атрибуты – это характеристики, присущие элементу вне контекста связей с другими объектами. Атрибут – аналог понятия «имя поля данных», используемого при проектировании баз данных. Перечень атрибутов, присущих тому или иному объекту или связи, определяется, исходя из роли объекта в решении задач ЕЦПНЗ.

Значение атрибута – конкретное значение данной характеристики, присущее данному объекту или связи. В качестве значения атрибута могут выступать текст, число, дата, формула, изображение и т. д.

Класс – это совокупность объектов, относящихся к данному подпространству, имеющих заданный набор атрибутов. В универсальном подпространстве выделим класс «Форматы», объекты которого описывают правила формирования значений атрибутов и связей всех объектов.

Связи – это вид «взаимоотношений» между парами объектов или значений атрибутов. Понятие связей в ЕЦПНЗ существенно шире аналогичных понятий, принятых в SKOS и OWL.

Связи ЕЦПНЗ подразделяются на три группы, каждая из которых относится к одному из типов – универсальному, квазиуниверсальному или специфическому.

Связи могут быть простыми и составными. Простые связи содержат (в терминах триплетов RDF [15]) указание на субъект, объект и (факультативно, в зависимости от конкретного вида связи) значение связи. Значения составных связей могут содержать «вложения» – иметь собственные атрибуты и их значения.

Универсальные связи являются простыми и указывают лишь на факт отношений между элементами и не зависят от классов объектов, которые они связывают. Они могут связывать любые элементы одного или нескольких классов. К связям этого типа относятся:

- «эквивалентно»;
- «пересекается»;
- «содержит»;
- «содержится в» (является частью, входит в состав).

Этот вид связей широко употребляется в предметных тезаурусах и при установлении соответствия между элементами классификационных систем. В ЕЦПНЗ он дополнительно используется при указании на соподчиненность подразделений организаций, на различные наименования организаций и публикаций, на различные написания фамилий и имен персон и т. п.

Квазиуниверсальные связи связывают субъекты различных классов с объектами заданного класса, они могут быть простыми или составными. Перечень квазиуниверсальных связей может пополняться по мере развития ЕЦПНЗ и добавления новых элементов. Примером квазиуниверсальных связей могут служить ссылки на статьи в энциклопедии или ссылки на предметные рубрики классификационных систем.

Специфические связи устанавливаются между субъектами и объектами заданных классов; они могут быть простыми и составными. Количество и вид специфических связей определяются при формировании онтологий конкретных классов. В отличие от универсальных связей, которые имеют статичный характер, у квазиуниверсальных связей, набор которых растет достаточно медленно, перечень специфических связей является достаточно динамичным, поскольку определяется развитием ЕЦПНЗ и возникающими перед ним задачами.

2. Справочники и словари

Для обеспечения процессов формирования контента ЕЦПНЗ, обработки запросов и навигации по ресурсам пространства необходимо иметь информацию о структуре элементов пространства и «взаимоотношениях» между ними. Эта информация содержится в соответствующих справочниках, которые формируются и дополняются администратором ЕЦПНЗ.

Справочники – структурированная информация, содержащая перечень и форматы представления элементов ЕЦПНЗ, связей между ними и их значениями. Структура справочников элементов ЕЦПНЗ определенного вида фиксирована и определяется элементами справочника CDSSK.

Справочники содержат информацию о том, что, куда и в каком виде вводить, какой и где реализовать формально-логический контроль при вводе данных, а также как связывать элементы запроса и различные характеристики объектов, в том числе, не присутствующие в явном виде в их атрибутах. Каждый элемент ЕЦПНЗ описывается в соответствующем справочнике. Каждый справочник в обязательном порядке содержит информацию о словарях значений атрибутов и связей, которые в нем указаны.

Словари значений атрибутов и связей содержат их конкретные значения. Каждое значение является уникальным, относится к одному из словарей и имеет свое имя (URN).

Словари объектов в качестве элементов содержат перечень URN значений атрибутов и связей, относящихся к конкретному объекту.

Словари «стандартных» значений атрибутов (таких как перечень ученых степеней, должностей, рубрики ГРНТИ или УДК и пр.) наполняются при первона-

чальной инсталляции ЕЦПНЗ, остальные словари наполняются в процессе формирования контента ЕЦПНЗ оператором ввода или программой пакетной загрузки данных.

3. Формализация описаний элементов ЕЦПНЗ

Каждый элемент ЕЦПНЗ имеет свое уникальное имя (URN), состоящее из имени справочника (или словаря), в который он входит, и порядкового номера элемента в справочнике (или словаре), отделенного от имени точкой. В свою очередь, имя справочника может быть элементом другого справочника, поэтому URN элемента может содержать различное число точек-разделителей. Значение элемента отделяется от его URN двоеточием и пробелом. Значения элементов справочников отделяются точкой с запятой и пробелом.

Для описания структуры справочников отдельных элементов ЕЦПНЗ (подпространств, классов, атрибутов и связей разного рода) предлагается унифицированный подход, основанный на формировании справочника верхнего уровня с именем CDSSK. Элемент CDSSK.1 описывает структуру справочников подпространств, элемент CDSSK.2 – справочников классов и т. д.

CDSSK.1: Структура справочника подпространств (ПП).

Справочник подпространств имеет имя SUBS; элемент справочника содержит три атрибута: наименование; код типа подпространства; описание подпространства. Код типа подпространства (далее – «префикс») состоит из двух символов; принимает значение UN для универсального подпространства и обозначается другими символами для тематического. Код может быть представлен двумя цифрами – кодом тематики верхнего уровня ГРНТИ или двумя буквами, если ТПП относится к более узкой тематике или содержит междисциплинарную информацию. Например, подпространству «Информатика» может быть присвоен префикс 20, подпространству «Вычислительная техника» префикс HW (от англ. «hardware»).

Имя справочника подпространств SUBS

Элемент справочника содержит 3 составляющих:

Наименование

Префикс ПП (2 символа)

Описание

Примеры:

SUBS.1: Универсальное; UN; подпространство, включающее классы объектов, не связанные непосредственно с конкретной научной тематикой, в том числе универсальные справочные данные.

SUBS.2: Информатика; 20; подпространство включает объекты, относящиеся к научному направлению «информатика»

CDSSK.2: Структура справочника класса объектов.

Класс объектов (Class). Определены два типа классов объектов – универсальные и локальные. Последние принадлежат какому-либо тематическому подпространству. Имя справочника классов: URN: Class.

Элемент справочника содержит 6 составляющих:

Наименование

Тип (универсальный – UN, локальный – LC)

Префикс (UNxy для универсального и <ПР>xy для локального, где <ПР> – префикс тематического подпространства–- два символа; xy – два буквенно-цифровых символа)

URN словаря атрибутов

URN словаря связей.

Описание

Примеры:

Class.1: персоны; UN; UNPS; A_UNPS; C_UNPS; информация о персонах, в той или иной мере связанных с научными исследованиями.

Class.16: Форматы представления данных; UN; UNFT; A_UNFT; C_UNFT; форматы представления атрибутов объектов и связей.

CDSSK.3: Структура справочника атрибутов.

Имя (URN) справочника атрибутов формируется в форме A_префикс класса.

Элемент справочника содержит 5 составляющих:

Наименование атрибута;

Формат представления значений атрибута (URN соответствующего элемента справочника объектов класса «Форматы данных»);

URN словаря значений атрибута (формируется в форме N_URN атрибута);

URN справочника связей значений атрибута (формируется в форме C_N_URN атрибута);

Дополнительная информация (пояснительный текст).

Пример (фрагмент справочника):

A_UNPS.1: фамилия; UNFT.10 [URN объекта из класса «Форматы данных», сообщающий, что атрибут является обязательным текстовым]; N_A_UNPS.1; C_N_A_UNPS.1 [в этом словаре связей содержатся указания на эквивалентность разных написаний фамилий]; фамилия выбирается из словаря, при отсутствии она вводится и проверяется на эквивалентность с другими написаниями;

A_UNPS.2: имя; UNFT.3 [URN объекта из класса «Форматы данных», сообщающий, что атрибут является необязательным текстовым]; N_A_UNPS.2; C_N_A_UNPS.2; имя выбирается из словаря, при отсутствии оно вводится и проверяется на эквивалентность с другими написаниями;

A_UNPS.3: отчество; UNFT.3; N_A_UNPS.3; ; отчество выбирается из словаря, при отсутствии оно вводится;

A_UNPS.4: дата рождения; UNFT.4 [URN объекта класса «Форматы данных», сообщающий, что элемент представляется в формате «гггг[мм[дд]]», является обязательным, уникальным]; N_A_UNTC.2 [ссылка на элемент словаря временных характеристик]; ; ;

CDSSK.4: Структура справочника универсальных связей.

Имя (URN) справочника: REUN.

Элемент справочника содержит 3 составляющих:

Наименование

URN значения словаря формата данных, определяющего форму представления данной связи

Описание связи

Пример:

REUN.1: Эквивалентность; N_A_UNFT.2.6; используется для обозначения идентичных атрибутов или связей (разные написания фамилий и имен, разные наименования одной организации, синонимы терминов и т. п.)

CDSSK.5: Структура справочника квазиуниверсальных связей.

Имя (URN) справочника: RQUN.

Элемент справочника содержит 6 составляющих:

Наименование

Префикс класса, являющегося «объектом связи»

Необходимость справочника значений (Y / N)

URN словаря значений (если указано Y) или пустое поле

URN значения словаря формата данных, определяющего форму представления данной связи

Описание связи.

Пример:

RQUN.4: Местоположение; UNPC; Y; N_A_UNPC.1; N_ A_UNFT.2.6: указывается местоположение объекта в виде, присутствующем в словаре географических наименований.

CDSSK.6: Структура справочника специфических связей.

Имя (URN) справочника: RESP.

Элемент справочник имеет «шапку» из 7-ми составляющих, которая в случае составной связи дополняется блоками, содержащими по 4 составляющих, описывающими иерархию значений связи.

Составляющие элементов справочника:

1.Наименование связи

2. Класс субъекта

3. Класс объекта

4. URN справочника атрибутов связи

5. URN словаря значений связи

6. Формат представления связи (URN значения элемента N_UNFT)

7.Количество подчиненных связей следующего уровня (0 - n)

Если не нуль, то добавляется блок связи второго уровня:

8. Наименование подчиненной связи 1

9. URN словаря атрибутов подчиненной связи 1

10. URN словаря значений подчиненной связи 1

11. Количество подчиненных связей следующего уровня (0 – n)

Если не 0, то определяется блок подчиненных связей третьего уровня, если 0, а в строке 7 n>1, определяется следующий блок связи второго уровня.

Пример:

RESP.5: UNPS; UNPB; связь персоны с публикацией; N_A_UNFT.2.5; A_RESP.5; 0;

CDSSK.7: Структура словаря значений атрибутов объектов и связей.

URN словаря формируется в форме N_URN атрибута. Элемент словаря имеет одну составляющую – значение в соответствии с форматом, URN которого указан в справочнике атрибутов.

Примеры:

N_A_UNPS.1.1: Менделеев

N_A_UNPS.4.1: N_A_UNTC.2.1

N_A_UNTC.2.1: 1834.12.08

CDSSK.8: Структура словаря связей

Имя словаря совпадает с URN справочника связи, указанном в соответствующем справочнике CDSSK.

Примеры:

пусть

N_A_UNPS.1.1: Андреев

N_A_UNPS.1.2: Andreev

N_A_UNPS.1.3: Andreyev,

тогда

REUN.1.1: <N_A_UNPS.1.1>< N_A_UNPS.1.2>

REUN.1.2: <N_A_UNPS.1.1>< N_A_UNPS.1.3>

Если персона с URN=UNPS.r является редактором и автором перевода публикации с URN=UNPB.s, то этот факт будет представлен двумя элементами словаря значений N_RESP.5:

N_RESP.5.n: < UNPS.r >< UNPB.s >=<N_A_RESP.5.2>

N_RESP.5.n+1: < UNPS.r >< UNPB.s >=<N_A_RESP.5.4>,

где элементы словаря N_A_RESP.5 представлены в виде:

N_A_RESP.5.2: редактор

N_A_RESP.5.4: автор перевода.

CDSSK.9: Структура словарей объектов.

Имя словаря совпадает с URN класса, к которому относится данный объект.

Элемент словаря представляет собой перечень URN элементов словарей атрибутов и связей, относящихся к данному объекту.

Элементы всех словарей формируются автоматически в процессе ввода данных в ЕЦПНЗ – либо программным путем (прикладная программа пакетного ввода данных обрабатывает справочники атрибутов и связей и записывает элементы в соответствующие словари), либо как результат диалога с оператором ввода. Во втором случае оператору предлагаются (на основе программной обработки справочников) наименования атрибутов вводимого объекта и связей с другими объектами. По каждым атрибуту и связи оператор должен выбрать уже имеющиеся в ЕЦПНЗ их значения или ввести новые с указанием значений всех необходимых связей.

4. Примеры формального описания объектов и связей

В настоящее время в универсальном подпространстве выделены классы объектов, которые условно разделены на две группы – предметные и вспомогательные. К предметным классам отнесены: «Персоны», «Публикации», «Квалификационные работы» «Документы», «Мультимедийные материалы», «Музейные предметы», «События», «Организации», «Политематические базы данных» «Награды». К вспомогательным: «Форматы данных», «Тезаурусы (предметные онтологии)», «Местоположение (географические характеристики)», «Временные характеристики», «Единицы измерения», «Научные направления», «Группы персон», «Числовые значения», «Языки», «Коллекции».

Для каждого класса объектов сформировано их формальное описание и предложен перечень атрибутов; для ряда предметных классов определены виды попарных специфических связей.

Рассмотрим несколько примеров.

4.1. Описание класса «форматы представления данных»

Class.16: Форматы представления данных; UN; UNFT; A_UNFT; C_UNFT; форматы представления атрибутов объектов и связей.

Каждый элемент словаря форматов UNFT содержит 6 атрибутов, определяемых элементами справочника A_UNFT, структура которого определена справочником CDSSK.5:

A_UNFT.1: тип представления данных; ; N_A_UNFT.1; ;используется для формально-логического контроля вводимых данных;

A_UNFT.2: вид формата; ; N_A_UNFT.2; ; используется при обработке данных;

A_UNFT.3: обязательное (r) или факультативное (f) значение атрибута; ; N_A_UNFT.3; ;используется для формально-логического контроля вводимых данных;

A_UNFT.4: уникальное (u) или множественное (m) значение атрибута; ; N_A_UNFT.4; ; используется для формально-логического контроля вводимых данных;

A_UNFT.5: ограничения по кодировке или структуре; ; N_A_UNFT.5; ; используется при формировании контента;

A_UNFT.6: ссылка на подробное описание формата; ; N_A_UNFT; ; используется в качестве справочного материала;

Значения атрибутов выбираются из соответствующих словарей.

Словари значений атрибутов, за исключением N_A_UNFT.3 и N_A_UNFT.4 пополняются по мере необходимости. Примеры элементов словарей:

N_A_UNFT.1.1: текст

N_A_UNFT.1.2: изображение

N_A_UNFT.1.3: видео

N_A_UNFT.1.5: любое число

N_A_UNFT.1.6: целое число

N_A_UNFT.1.7: дата в формате гггг[.мм[.дд]]

N_A_UNFT.1.8: время в формате чч[.мм[.сек]]

N_A_UNFT.1.9: время в формате гггг.мм.дд. чч[.мм[.сек]]

N_A_UNFT.1.10: связи

N_A_UNFT.2.1: TEX

N_A_UNFT2.2: PDF

N_A_UNFT.2.3: таблицы Excel, csv

N_A_UNFT.2.4: простая связь первого типа между объектами, атрибутами или значениями O1 и O2, она описывается «простым триплетом» вида <URNc>:<URNO1><URNO2>, где URNc – URN конкретной связи. Примеры: фамилия «Петров» эквивалентна «Petrov»; статья входит в состав энциклопедии; организация включает подразделение и т. п.

N_A_UNFT.2.5: простая связь второго типа, указывающая на субъект, объект, URN связи и URN значения связи. Формат представления связи имеет вид: <URNc>:<URN субъекта><URN объекта>=<URN элемента словаря значений соответствующего атрибута связи>. Пример: персона P1 является сотрудником организации O1 (атрибут специфической связи «персона – «организация»1) в должности инженера (значение атрибута).

N_A_UNFT.2.6: составная связь третьего типа – «многоуровневый триплет»– случай, когда у значения атрибута связи имеются свои атрибуты с соответствующими значениями, у значений имеются атрибуты, каждый из которых, в свою очередь, имеет свое значение; Формат представления связи имеет вид: <URN субъекта><URN объекта> <URNc> = <URN элемента словаря значений соответствующего атрибута связи> <URN атрибута элемента словаря значений> = <URN значения атрибута>. Пример: персона P1 является сотрудником организации O1, работает в должности инженера с такой-то даты

<URN P1> <URN O1><URNc>=<URN значения «сотрудник»><URN атрибута значения «должность>=<URN значения «инженер>»><URN атрибута «начало работы»>=<URN значения даты>.

N_A_UNFT.2.9: Составная связь четвертого типа – «древовидный триплет», используется в случаях, когда у одного значения атрибута связи может быть несколько атрибутов со своими значениями, у каждого из которых могут быть свои атрибуты со своими значениями и т. д. Формат представления связи имеет вид: <URN субъекта><URN объекта> <URNc> = <URN элемента словаря значений соответствующего атрибута связи> [[блок 1 [блок 1.1 [блок 1.1.1.] [блок 1.1.2]] блок 2

[1] Другими атрибутами могут быть «спонсор», «учредитель», акционер и т. п.

[блок 2.1] и т. д.]], где блок представляет собой структуру <URN атрибута значения связи i-того уровня>=<URN одного из значений этого атрибута>

N_A_UNFT.2.10: алгоритмы контроля 10-значного номера ISBN. Если ISBN=N_1 N_2 … N_{10}, то $N_{10} = 11 - \left(S - 11*[S/11]\right)$, где $S = \sum_{i=1}^{9} i * N_i$, [S/11] – целая часть результата деления S на 11, если при вычислении N_{10} оказывается равным десяти, оно записывается римским числом X.

N_A_UNFT.2.11: алгоритмы контроля 13-значного номера ISBN. Если ISBN=N_1 N_2 ... N_{13}, то $N_{13} = 10 - \left(R - 10*[R/10]\right)$, где $R = \sum_{i=0}^{6} N_{2i+1} + 3\sum_{j=1}^{6} N_{2j}$, [R/10] – целая часть результата деления R на 10.

Третий и четвертый атрибуты справочника форматов принимают одно из двух значений:

N_A_UNFT.3.1: r

N_A_UNFT.3.2: f

N_A_UNFT.4.1: u

N_A_UNFT.4.2: m

Пример словаря значений атрибута «ограничения по кодировке или структуре».

N_A_UNFT.5.1: JPG

N_A_UNFT.5.2: MP4

N_A_UNFT.5.3: UniCode UTF-8

N_A_UNFT.5.4: арабские цифры

Словарь значений атрибута «ссылка на подробное описание формата»:

N_A_UNFT.6.1: https://habr.REm/ru/post/454944/

N_A_UNFT.6.2: https://open-file.ru/types/mp4

N_A_UNFT.6.3: https://ru.wikipedia.org/wiki/Юникод

N_A_UNFT.6.4: https://ru.wikipedia.org/wiki/Коды_языков

Примеры конкретных элементов справочника форматов, используемые при описаниях структур других справочников:

Текст, только буквы, в кодировке UniCode UTF-8, атрибут обязательный, значение уникальное

UNFT.1: N_A_UNFT.1.1; ; N_A_UNFT.3.1; N_A_UNFT.4.1; N_A_UNFT.5.3; N_A_UNFT.6.3;

Любой текст, атрибут обязательный, значение уникальное

UNFT.2: N_A_UNFT.1.1; ; N_A_UNFT.3.1; N_A_UNFT.4.1;;;

Текст, только буквы, атрибут необязательный, значение множественное

UNFT.3: N_A_UNFT.1.2; ;N_A_UNFT.3.2; N_A_UNFT.4.1; ; ;

Формат описания связей типа <URN субъекта> <URN связи> <URN объекта>

UNFT.4: N_A_UNFT.1.10; N_A_UNFT.2.4; ; ; ; ;

Дата в формате гггг[.мм[.дд]], атрибут необязательный, значение уникальное

UNFT.5: N_A_UNFT.1.8; ;N_A_UNFT3.2; N_A_UNFT.4.1; ; ;

Любой текст, атрибут необязательный, значение уникальное

UNFT.6: N_A_UNFT.1.1; ; N_A_UNFT.3.2; N_A_UNFT.4.1;;;

4.2. Описание класса «персоны»

Class.1: Персоны; UN; UNPS; A_UNPS; C_UNPS; информация о персонах, в той или иной мере связанных с научными исследованиями;

Справочник персон будет иметь имя UNPS, а конкретные объекты, входящие в этот класс, будет иметь URN=UNPS.k.

Объект класса «персоны» в ЕЦПНЗ идентифицируется значениями атрибутов, перечисленных в справочнике A_UNPS, структура которого описана в справочнике CDSSK.5. По мере необходимости он может дополняться новыми элементами, что не нарушит существовавшую до этого структуру. Значения атрибутов содержатся в словарях, указанных в соответствующих элементах справочника. Пример элементов справочника атрибутов объектов класса «Персоны»:

A_UNPS.1: фамилия; UNFT.1; N_A_UNPS.1; C_N_A_UNPS.1; фамилия выбирается из словаря, при отсутствии она вводится и проверяется на эквивалентность с другими написаниями;

A_UNPS.2: имя; UNFT.1; N_A_UNPS.2; C_N_A_UNPS.2; имя выбирается из словаря, при отсутствии оно вводится и проверяется на эквивалентность с другими написаниями;

A_UNPS.3: отчество; UNFT.3; N_A_UNPS.3; ; отчество выбирается из словаря, при отсутствии оно вводится;

A_UNPS.4: дата рождения; UNFT.5; N_A_UNTC.2 [URN словаря значений соответствующего атрибута объектов класса «временные характеристики»]; ; ;

A_UNPS.5: место рождения; UNFT.3; UNGC [URN словаря объектов класса «местонахождение»]; ; ;

A_UNPS.6: дата смерти; UNFT.5.; N_A_UNTC.2; ; ;

A_UNPS.7: место смерти; UNFT.3; UNGC; ; ;

A_UNPS.8: квалификация (ученая степень); UNFT.3; N_A_UNPS.8

A_UNPS.9: ученое звание; UNFT.3; N_A_UNPS.9; ; ;

A_UNPS.10: биография; UNFT.2; N_A_UNPS.10; ; ;

A_UNPS.11: библиография персоны; UNFT.6; N_A_UNPS.11; ; ;

A_UNPS.12: библиография о персоне; UNFT.6; N_A_UNPS.12; ; ;

Элементы словарей N_A_UNPS.8 и N_A_UNPS.9 заполняются на административном уровне на основе существующих градаций ученых степеней и званий. Словарь местонахождений UNGC может быть также заполнен данными из имеющихся географических информационных систем и дополняться по мере необходимости. Остальные словари заполняются данными, относящимися к конкретным персонам, по мере наполнения ЕЦПНЗ.

Дополнительные характеристики персон описываются как связи с другими классами объектов. В частности, идентификаторы авторов в российских и международных системах представляются как связи с объектами класса «политематические базы данных». Рассмотрим, в качестве примера, структуру связи персоны с публикацией.

Связь персоны с публикацией является простой связью второго типа, описываемой форматом N_A_UNFT.2.5. Она может принимать несколько значений (персона может быть автором и художником издания, одним из авторов и редакторов и т. п.). Обозначим эту связь как RESP.5.

Справочник этой связи будет иметь вид:

RESP.5: UNPS; UNPB; связь персоны с публикацией; N_A_UNFT.2.5; A_RESP.5; 0;

Справочник атрибутов представляется в виде:

A_RESP.5.1: Роль персоны в создании публикации; UNFT.i; N_A_RESP.5; ; ;

Второй элемент (UNFT.i) указывает, что значение атрибута содержит только буквы, является обязательным, и одной персоне может соответствовать несколько его значений.

Словарь возможных значений атрибута (дополняется по мере необходимости):

N_A_RESP.5.1: автор

N_A_RESP.5.2: редактор

N_A_RESP.5.3: составитель

N_A_RESP.5.4: автор перевода

N_A_RESP.5.5: художник

N_A_RESP.5.6: о нем

N_A_RESP.5.7: владелец авторских прав

Пример конкретного значения – персона с URN=UNPS.r является редактором и автором перевода публикации с URN=UNPB.s:

N_RESP.5.n: < UNPS.r >< UNPB.s >=<N_A_RESP.5.2>

N_RESP.5.n+1: < UNPS.r >< UNPB.s >=<N_A_RESP.5.4>.

Итоговое представление данных о конкретной персоне и ее связях с другими объектами универсального и тематических подпространств представляется в виде строки словаря, содержащей последовательность URN значений словарей атрибутов (N_A_UNPS.i.j), последовательность URN значений связей между персонами и другими объектами (N_RESP.n.m).

UNPS.i: N_A_UNPS.1.a; N_A_UNPS.2.b;…;N_A_UNPS.12.k; N_RESP.i.j;…;N_RESP.q.z

Аналогично представляются и другие объекты. Совокупность словарей объектов и словарей значений связей представляет собой замкнутую систему, внутри которой, используя мнемонику формирования справочников разного уровня, можно реализовать многоаспектный поиск данных и навигацию между разнородными элементами.

5. Заключение

Предложенная структура онтологии ЕЦПНЗ в настоящее время моделируется на примере развития электронной библиотеки «Научное наследие России» (ЭБ ННР) [16]. Библиотека поддерживает такие классы объектов, как «персоны»,

«публикации», «музейные объекты», «коллекции». Традиционный поисковый интерфейс позволял искать объекты определенного класса по заданным значениям их атрибутов с возможностью использования булевой логики. Реализованная в последней версии ЭБ опция «расширенный поиск» позволяет искать объекты не только по заданным значениям их атрибутов, но и по значению связей с объектами другого класса. Например, пользователь имеет возможность найти публикации, в которых персоны играли роль не только авторов, но и редакторов или переводчиков; найти музейные объекты, для которых персоны выступали в роли «автора сбора»; найти публикации, связанные с музейными объектами, и т. п.

В плане развития исследований предполагается расширить модельную базу путем постепенного добавления новых классов объектов и связей.

Работы выполняются в МСЦ РАН – филиале ФГУ ФНЦ НИИСИ РАН в рамках государственного задания по теме FNEF-2023-0014.

СПИСОК ЛИТЕРАТУРЫ

1. *Савин Г.И.* Единое цифровое пространство научных знаний: цели и задачи // Информационные ресурсы России. 2020. № 5. С. 3–5. https://doi.org/10.51218/0204-3653-2020-5-3-5

2. *Антопольский А.Б.и др.* Принципы построения и структура единого цифрового пространства научных знаний (ЕЦПНЗ) // Научно-техническая информация. Сер. 1. 2020. № 4. С. 9–17. https://doi.org/10.36535/0548-0019-2020-04-2.

3. *Каленов Н.Е., Сотников А.Н.* Архитектура единого цифрового пространства научных знаний // Информационные ресурсы России. 2020. № 5. С. 5–8. https://doi.org/10.51218/0204-3653-2020-5-5-8

4. *Атаева О.М., Каленов Н.Е., Серебряков В.А.* Онтологический подход к описанию единого цифрового пространства научных знаний // Электронные библиотеки. 2021. Т. 24, № 1. С. 3–19. https://doi.org/10.26907/1562-5419-2021-24-1-3-19

5. *Каленов Н.Е., Серебряков В.А.* Об онтологии Единого цифрового пространства научных знаний // Информационные ресурсы России. 2020. № 5. С. 10–12. https://doi.org/10.51218/0204-3653-2020-5-10-12

6. W3C 2009. SKOS Simple Knowledge Organization System Reference. W3C Recommendation 18 August 2009. URL: https://www.w3.org/TR/skos-reference/ (дата обращения: 10.01.2023).

7. SKOS Simple Knowledge Organization System. URL: http://www.w3.org/TR/skos-reference/#xl-Label (дата обращения: 10.01.2023).

8. Web Ontology Language (OWL). URL: https://www.w3.org/OWL/ (дата обращения: 10.01.2023).

9. *Marcia Lei Zeng & Philipp Mayr.* Knowledge Organization Systems (KOS) in the Semantic Web: a multi-dimensional review // International Journal on Digital Libraries. 2018. URL: https://arxiv.org/pdf/1801.04479.pdf/ (дата обращения: 10.01.2023).

10. *Pattuelli M. Cristina, Alexandra Provo, and Hilary Thorsen 2*015. Ontology building for Linked Open Data: A pragmatic perspective. Journal of Library Metadata. 2015. Vol. 15, No. 3-4. P. 265–294.

11. *Volkan Çağdaş and Erik Stubkjær*. A SKOS vocabulary for Linked Land Administration: Cadastre and Land Administration Thesaurus. Land Use Policy. 2015. Vol. 49. P. 668–679.

12. *Zapilko Benjamin, Johann Schaible, Philipp Mayr, and Brigitte Mathiak.* TheSoz: A SKOS representation of the Thesaurus for the Social Sciences. Semantic Web Journal (SWJ). 2013. Vol. 4, No. 3. P. 257–263.

13. *Zeng Marcia Lei*. Create micro thesauri and other datasets from the Getty LOD vocabularies. In MW17: Museums and the Web Conference, April 19–22, 2017 Cleveland, Ohio, USA. URL: http://www.getty.edu/research/tools/vocabularies/zeng_microthesauri_getty_lod.pdf (дата обращения: 10.01.2023)

14. Ontolog-Forum. URL: https://groups.google.REm/forum/#!forum/ gettyvocablod (дата обращения: 10.01.2023).

15. Resource Description Framework (RDF): Concepts and Abstract Syntax. URL: https://clck.ru/gwVBC (дата обращения: 10.01.2023).

16. Электронная библиотека «Научное наследие России». URL: http://heritage1.jscc.ru/ (дата обращения: 10.01.2023).

UNIFIED REPRESENTATION OF THE COMMON DIGITAL SPACE OF SCIENTIFIC KNOWLEDGE ONTOLOGY

N. Kalenov[1] [0000-0001-5269-0988], A. Sotnikov[2] [0000-0002-0137-1255]

[1, 2]Joint SuperComputer Center of the Russian Academy of Sciences – Branch of Federal State Institution "Scientific Research Institute for System Analysis of the Russian Academy of Sciences"

[1]nkalenov@jscc.ru, [2]asotnikov@jscc.ru

Abstract

The Common Digital Space of Scientific Knowledge (CDSSK) is a digital information environment aggregating heterogeneous information related to various aspects of scientific knowledge. One of the important functions of the CDSSK is to provide information for solving artificial intelligence problems, which makes it necessary to support data in a structure that complies with the rules of the semantic WEB. The features of the CDSSK are, on the one hand, the polythematics and heterogeneity of content elements, on the other hand, the high dynamics of the emergence of new types of objects and connections between them, which is due to the specifics of the development of science. At the same time, it should be possible to navigate through heterogeneous space resources using semantic links between them. The possibilities of the CDSSK are largely determined by the structure of the ontology of space, the model of which is proposed in this paper. Within the framework of the model, the hierarchical structuring of the CDSSK ontology is carried out; such elements as "subspace", "class of objects", "object", "attributes of an object", three types of pairwise relations of objects or attributes (universal, quasi-universal and specific) are distinguished and defined. The structure of each elements type is determined by a "reference book" of a unified type; specific values of attributes and relationships are contained in dictionaries of a unified structure. A class of "Formats" objects describing the rules for the formation of attributes and values of relationships is allocated. The formalization of CDSSK reference books and dictionaries representations is proposed. The proposed model allows you to simply add new types of objects, of their pairwise relationships and attributes to the space, as needed.

Keywords*: digital space of scientific knowledge, ontologies, structuring, related data, data attributes, semantic WEB.*

REFERENCES

1. Savin G.I. Yedinoye tsifrovoye prostranstvo nauchnykh znaniy: tseli i zadachi // Informatsionnyye resursy Rossii. 2020. № 5. S. 3–5. https://doi.org/0.51218/0204-3653-2020-5-3-5

2. *Antopol'skiy A.B. i dr.* Printsipy postroyeniya i struktura Edinogo tsifrovogo prostranstva nauchnykh znaniy // Nauchno-tekhnicheskaya informatsiya. ser. 1. 2020. № 4. S. 9–17. https://doi.org/10.36535/0548-0019-2020-04-2

3. *Kalenov N.Ye., Sotnikov A.N.* Arkhitektura shirokogo rasprostraneniya nauchnykh znaniy // Informatsionnyye resursy Rossii. 2020. № 5. S. 5–8. https://doi.org/10.51218/0204-3653-2020-5-5-8

4. *Atayeva O.M., Kalenov N.Ye., Serebryakov V.A.* Ontologicheskiy podkhod k opisaniyu obshchedostupnykh nauchnykh prostranstv // Elektronnyye biblioteki. 2021. T. 24, № 1. S. 3–19. https://doi.org/10.26907/1562-5419-2021-24-1-3-19

5. *Kalenov N.Ye., Serebryakov V.A.* Ob ontologii Yedinogo otkrytogo prostranstva nauchnykh znaniy // Informatsionnyye resursy Rossii. 2020. № 5. S. 10–12. https://doi.org/10.51218/0204-3653-2020-5-10-12.eller

6. W3C 2009. SKOS Simple Knowledge Organization System Reference. W3C Recommendation 18 August 2009. https://www.w3.org/TR/skos-reference/ (accessed 10.01.2023).

7. SKOS Simple Knowledge Organization System. URL: http://www.w3.org/TR/skos-reference/#xl-Label (accessed: 10.01.2023).

8. Web Ontology Language (OWL). URL: https://www.w3.org/OWL/ (accessed: 10.01.2023).

9. *Marcia Lei Zeng & Philipp Mayr.* Knowledge Organization Systems (KOS) in the Semantic Web: a multi-dimensional review // International Journal on Digital Libraries. 2018. URL: https://arxiv.org/pdf/1801.04479.pdf/ (accessed: 10.01.2023).

10. *Pattuelli M. Cristina, Alexandra Provo, and Hilary Thorsen 2*015. Ontology building for Linked Open Data: A pragmatic perspective. Journal of Library Metadata. 2015. Vol. 15, No. 3-4. P. 265–294.

11. *Volkan Çağdaş and Erik Stubkjær*. A SKOS vocabulary for Linked Land Administration: Cadastre and Land Administration Thesaurus. Land Use Policy. 2015. Vol. 49. P. 668–679.

12. *Zapilko Benjamin, Johann Schaible, Philipp Mayr, and Brigitte Mathiak.* TheSoz: A SKOS representation of the Thesaurus for the Social Sciences. Semantic Web Journal (SWJ). 2013. Vol. 4, No. 3. P. 257–263.

13. *Zeng Marcia Lei*. Create micro thesauri and other datasets from the Getty LOD vocabularies. In MW17: Museums and the Web Conference, April 19–22, 2017 Cleveland, Ohio, USA.
URL: http://www.getty.edu/research/tools/vocabularies/zeng_microthesauri_getty_lod.pdf (accessed: 10.01.2023)

14. Ontolog-Forum. URL: https://groups.google.REm/forum/#!forum/gettyvocablod (accessed: 10.01.2023).

15. Resource Description Framework (RDF): Concepts and Abstract Syntax. URL: https://clck.ru/gwVBC (accessed: 10.01.2023).

16. Elektronnaya biblioteka “Nauchnoe nasledie Rossii”.
URL: http://heritage1.jscc.ru/ (accessed: 10.01.2023).

СВЕДЕНИЯ ОБ АВТОРАХ

КАЛЕНОВ Николай Евгеньевич – главный научный сотрудник Межведомственного Суперкомпьютерного Центра РАН – филиала Федерального государственного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук», доктор технических наук, профессор.

Nikolay Evgenievich KALENOV – Chief Researcher of the Joint SuperComputer Center of the Russian Academy of Sciences – Branch of the Federal State Institution «Scientific Research Institute for System Analysis of the Russian Academy of Sciences», Doctor of Technical Sciences, Professor.

email: nekalenov@mail.ru;

ORCID: 0000-0001-5269-0988

СОТНИКОВ Александр Николаевич – заместитель директора Межведомственного Суперкомпьютерного Центра РАН – филиала Федерального государственного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук», доктор физико-математических наук, профессор.

Alexander Nikolaevch SOTNIKOV – Deputy Director of the Joint SuperComputer Center of the Russian Academy of Sciences – Branch of the Federal State Institution “Scientific Research Institute for System Analysis of the Russian Academy of Sciences”, Doctor of Sciences (Math), Professor.

email: asotnikov@jscc.ru;

ORCID: 0000-0002-0137-1255

Материал поступил в редакцию 10 января 2023 года

УДК 004.05

ОПЫТ ВЕРИФИКАЦИИ РЕАЛИЗАЦИЙ КЛИЕНТА ПРОТОКОЛА TLS 1.3

А. В. Никешин[1] [0000-0001-5781-9736], В. З. Шнитман[2] [0000-0002-1509-0972]

[1, 2]Институт системного программирования им. В.П. Иванникова РАН, ул. А. Солженицына, 25, г. Москва, 109004

[1]alexn@ispras.ru, [2]vzs@ispras.ru

Аннотация

Представлен опыт верификации реализаций клиента криптографического протокола TLS версии 1.3. TLS сегодня является одним из наиболее востребованных криптографических протоколов, предназначенных для создания защищенных каналов передачи данных. Протокол обеспечивает необходимую для своих задач функциональность: конфиденциальность передаваемых данных, целостность данных, аутентификацию сторон. В новой версии протокола TLS 1.3 была существенно переработана архитектура, устранен ряд недостатков предыдущих версий, выявленных как при разработке реализаций, так и в процессе их эксплуатации.

В работе использован новый тестовый набор для верификации реализаций клиента протокола TLS 1.3 на соответствие спецификациям интернет, разработанный на основе спецификации RFC 8446 с использованием технологии UniTESK и методов мутационного тестирования. Для тестирования реализаций на соответствие формальным спецификациям применена технология UniTESK, предоставляющая средства автоматизации тестирования на основе использования конечных автоматов. Состояния тестируемой системы задают состояния автомата, а тестовые воздействия – переходы этого автомата. При выполнении перехода заданное воздействие передается на тестируемую реализацию, после чего регистрируются реакции реализации и автоматически выносится вердикт о соответствии наблюдаемого поведения спецификации. Мутационные методы тестирования используются для обнаружения нестандартного поведения тестируемой системы (завершение из-за фатальной ошибки, «подвисание», ошибки доступа к памяти) с помощью передачи некорректных данных, такие ситуации часто остаются за рамками

требований спецификаций. В сообщения, сформированные на основе разработанной модели протокола, вносятся какие-либо изменения. Модель протокола дает возможность вносить изменения в поток данных на любом этапе сетевого обмена, что позволяет тестовому сценарию проходить через все значимые состояния протокола и в каждом таком состоянии проводить тестирование реализации в соответствие с заданной программой. Представленный подход доказал свою эффективность в нескольких наших проектах при тестировании сетевых протоколов, обеспечив обнаружение различных отклонений от спецификации и других ошибок. Текущая работа является частью проекта верификации протокола TLS 1.3 и охватывает реализации клиентской части протокола.

***Ключевые слова**: безопасность, TLS, TLSv1.3, протоколы, тестирование, оценка устойчивости, интернет, стандарты, формальные методы спецификации.*

ВВЕДЕНИЕ

Новая версия протокола TLS 1.3, представленная в августе 2018 года, постепенно вытесняет устаревшие версии [1, 2]. Многие разработчики программного обеспечения (ПО) уже включили поддержку TLS 1.3 в свои реализации. Поэтому исследования в области верификации и безопасности реализаций новой версии протокола остаются актуальной задачей, решающей несколько важных задач: проверка функциональной совместимости различных реализаций, проверка соответствия реализаций требованиям спецификации и устойчивость реализаций к нестандартным воздействиям.

В новой версии протокола оптимизирована структура обмена сообщениями, последние сгруппированы в несколько блоков: обмен ключами, параметры сервера, фаза аутентификации. Сделан акцент на шифрование как можно большего количества данных. Часть механизмов протокола перенесена в расширения, а набор сервисов безопасности расширен.

Предыдущие работы были посвящены верификации серверных реализаций протокола TLSv1.3 [3,4]. Текущая работа является ее продолжением и охватывает реализации клиентской части протокола.

В настоящей работе использованы наработанные нами методики тестирования сетевых протоколов: автоматизированное тестирование на соответствие формальным спецификациям и методы мутации данных, доказавшие свою эффективность в наших предыдущих проектах при тестировании сетевых протоколов.

Структура статьи: в первом разделе представлен краткий обзор новой версии протокола, во втором разделе обозначены применяемые методы верификации, в третьем разделе описан тестовый стенд, в четвертом разделе представлены результаты тестирования.

1. КРАТКИЙ ОБЗОР НОВОЙ ВЕРСИИ ПРОТОКОЛА

Напомним общую архитектуру протокола.

TLS предназначен для создания защищенных каналов передачи данных, обеспечивая их конфиденциальность и целостность, а также аутентификацию сторон. Механизм обработки сообщений состоит из двух уровней:

– Протокол рукопожатия (Handshake protocol) является основной частью архитектуры TLS. Он выполняет аутентификацию взаимодействующих сторон, согласовывает версию протокола, криптографические режимы и параметры безопасности, устанавливает общий ключевой материал. Встроенные механизмы защиты противодействуют вмешательству третьей стороны и понижению уровня безопасности согласованного криптографического набора.

– Протокол записей (Record protocol) используется для инкапсуляции протоколов более высокого уровня, в том числе протокола рукопожатия. Он использует параметры, установленные протоколом рукопожатия для защиты данных, передаваемых между партнерами. Весь трафик делится на последовательность записей, каждая из которых независимо защищается с помощью согласованных криптографических ключей. Протокол записей работает поверх надежного транспортного протокола (как правило, TCP) и не зависит от протоколов прикладного уровня.

Далее представлено полное рукопожатие TLSv1.3:

C→S: ClientHello (Key Exchange phase)
+ key_share*
+ signature_algorithms*

```
                + psk_key_exchange_modes*
                + pre_shared_key*
C←S:        ServerHello
                + key_share*
                + pre_shared_key*
               {EncryptedExtensions}          (Server Parameters phase)
               {CertificateRequest*}
               {Certificate*}                 (Authentication phase)
               {CertificateVerify*}
               {Finished}
               [Application Data*]
C→S:
               {Certificate*}
               {CertificateVerify*}
               {Finished}
[Application Data]  <-------->  [Application Data]
```

* Обозначает необязательные или зависящие от ситуации сообщения/расширения, которые посылаются не всегда.

{} Обозначает сообщения, защищенные с помощью сеансовых ключей для обмена рукопожатия.

[] Обозначает сообщения, защищенные с помощью сеансовых ключей для прикладных данных.

Как видно из схемы обмена, сообщения сгруппированы в три блока:

Первый блок предназначен для обмена ключами и состоит из двух сообщений: ClientHello и ServerHello. Клиент, начиная диалог, отправляет ClientHello, в которое включает наборы поддерживаемых алгоритмов и соответствующий ключевой материал. Сервер выбирает из предложенного множества один криптографический набор, который соответствует его политикам безопасности, добавляет свой ключевой материал и отправляет эти данные клиенту. Здесь применяются встроенные механизмы защиты от понижения согласованного номера версии протокола TLS, а сам процесс согласования версий перемещен в расширения, что

должно улучшить совместимость с существующими серверами и промежуточными устройствами, которые часто неправильно реализуют согласование версий.

Теперь сервер обладает полным набором данных, чтобы вычислить все необходимые сеансовые ключи. Потому все последующие сообщения (в отличие от предыдущих версий TLS) отправляются в зашифрованном виде. Кроме этого, спецификация разрешает серверу сразу начать отправку прикладных данных, не дожидаясь получения ответа с аутентификацией клиента. Такая схема обмена позволяет быстрее начать передачу пользовательских данных за счет уменьшения количества служебных сообщений (в предыдущей версии TLS присутствовало дополнительное обязательное сообщение ChangeCipherSpec, и требовался дополнительный раунд обмена). Однако у нее есть и обратная сторона. Во-первых, в этой точке обмена любые данные посылаются не аутентифицированному клиенту. Во-вторых, появляется возможность проведения эффективных DOS-атак (Denial of Service, отказ в обслуживании), поскольку затраты на серию сообщений ClientHello небольшие, а сервер в ответ на каждое такое сообщение вынужден вычислять полный набор ключей и хэш-суммы сообщений CertificateVerify и Finished (при этом никакой аутентификации от клиента не требуется).

Второй блок сообщений передает параметры сервера. Сообщение EncryptedExtensions содержит расширения, ответные для соответствующих расширений ClientHello, которые не требуются для создания криптографических параметров (например, поддержка протокола прикладного уровня). Такие расширения ранее отправлялись в открытом виде. Также сервер может запросить сертификат клиента.

Третий блок сообщений – фаза аутентификации. Сообщения Certificate, CertificateVerify, Finished выполняют аутентификацию сервера (и, при необходимости, клиента), подтверждают созданные сеансовые ключи и обеспечивают целостность всего обмена рукопожатия.

Для случаев заранее распределенных ключей (PSK, pre-shared key) TLS 1.3 позволяет использовать сокращенный режим рукопожатия, в нем используется меньше сообщений для обмена и трудоемких вычислений с ключами, что может быть полезно в условиях отсутствия инфраструктуры для работы с сертификатами.

```
C→S:    ClientHello
         + key_share*
         + psk_key_exchange_modes
         + pre_shared_key
C←S:    ServerHello
         + pre_shared_key
         + key_share*
        {EncryptedExtensions}
        {Finished}
        [Application Data*]
C→S:
        {Finished}

[Application Data]  <-------> [Application Data]
```

Режим PSK можно использовать в двух вариантах, определяемых расширением psk_key_exchange_modes: PSK и PSK-(EC)DHE. Последний дополняет процедуру формирования сеансовых ключей из PSK алгоритмом Диффи-Хеллмана (используется расширение key_share), что увеличивает как надежность итоговых ключей (обеспечивая прямую секретность, forward secrecy), так и сложность дополнительных криптографических вычислений.

Из других особенностей новой версии протокола TLS можно отметить механизм возобновления сессии. В предыдущей версии для этого существовал специальный режим возобновления сеанса. Теперь же используются общий ключ, согласованный в предыдущем обмене рукопожатия, и указанный выше обмен PSK.

На основе режима рукопожатия PSK реализован новый для протокола TLS способ отправки так называемых «ранних данных» (early data, 0-RTT). При наличии общего ключа PSK TLS 1.3 позволяет клиенту отправить некоторые данные во время первого рейса. Клиент использует PSK одновременно для аутентификации сервера и шифрования ранних данных.

```
C→S:    ClientHello
         + early_data
         + key_share*
         + psk_key_exchange_modes
         + pre_shared_key
        (Application Data*)
C←S:    ServerHello
         + pre_shared_key
         + key_share*
        {EncryptedExtensions}
         + early_data*
        {Finished}
        [Application Data*]
C→S:
        (EndOfEarlyData)
        {Finished}

    [Application Data]  <-------> [Application Data]
```

Данные 0-RTT добавляются к рукопожатию 1-RTT в первом рейсе в сообщение ClientHello. Однако свойства безопасности для таких данных слабее, чем для других данных TLS, например, нет защиты от повторного воспроизведения между разными соединениями (в спецификации рассмотрены такие виды атак), хотя внутри одного соединения данные 0-RTT дублироваться не могут, т. к. данные 0-RTT и 1-RTT защищены разными ключами.

TLSv1.3 как расширяемый протокол предоставляет дополнительные, необязательные сервисы безопасности, согласуемые, как правило, через механизм расширений. Среди них можно отметить несколько наиболее важных.

Это механизм обновления сеансовых ключей, повышающий безопасность передачи данных; никаких дополнительных данных передавать не требуется. Сообщение KeyUpdate просто сообщает партнеру, что отправитель создал новые сеансовые ключи (используется предыдущий ключевой материал), и последующие сообщения зашифрованы новыми ключами.

Расширение ServerName позволяет клиенту указать имя сервера, с которым он связывается [5]. Это может пригодиться, если по одному сетевому адресу размещено несколько «виртуальных» серверов, для каждого из которых используется свой сертификат. Для сервера получение этого расширения носит рекомендательный характер, при этом должны учитываться и другие настройки политик безопасности.

Расширение MaxFragmentLength позволяет клиенту использовать сообщения меньшего размера, чем предусмотрено спецификацией (2^{14} байт) [5]. Однако у него есть ряд существенных недостатков, связанных с фиксированным набором неизменяемых значений. Расширение RecordSizeLimit решает эти проблемы, позволяя задать произвольное максимальное значение сообщений (не превышающее значение, определенное версией протокола, согласованной партнерами) для каждого направления передачи данных. В спецификации более подробно изложены преимущества и недостатки этих двух расширений [6].

Расширение PostHandshakeAuth указывает серверу, что клиент хочет выполнить аутентификацию после рукопожатия [2]. В этом случае сервер может запросить аутентификацию клиента в любой момент времени после завершения рукопожатия путем посылки сообщения CertificateRequest. Клиент должен ответить соответствующими сообщениями аутентификации.

C←S: [CertificateRequest]

C→S:

[Certificate]

[CertificateVerify*]

[Finished]

Если у клиента нет соответствующих сертификатов, то отправляется пустое сообщение Certificate, а сообщение CertificateVerify не используется.

Сервер может отправить несколько сообщений CertificateRequest, как в разное время, так и последовательно (например, если требуется доступ к нескольким сервисам). При этом ответы могут приходить в произвольной последовательности (для разделения запросов используются соответствующие уникальные идентификаторы).

Такая функциональность протокола может также использоваться и для режима рукопожатия PSK, во время которого сертификаты не используются, но зато после завершения рукопожатия можно запросить сертификат клиента (если ранее клиент включил расширение "post_handshake_auth" в сообщение ClientHello).

2. ВЕРИФИКАЦИЯ ПРОТОКОЛА

В процессе тестирования сетевых протоколов решается несколько важных задач: проверяются функциональная совместимость различных реализаций, соответствие реализации требованиям спецификации и устойчивость реализации к нестандартным воздействиям.

В наших проектах используются наработанные нами методики по тестированию сетевых протоколов: автоматизированное тестирование на соответствие формальным спецификациям и методы мутации данных.

В текущих экспериментах использована модель протокола TLS версии 1.3, разработанная нами на основе спецификаций RFC и описывающая сложную схему функционирования протокола.

Для тестирования реализаций на соответствие формальным спецификациям применена технология UniTESK, предоставляющая средства автоматизации тестирования на основе использования конечных автоматов [7]. Состояния тестируемой системы задают состояния автомата, а тестовые воздействия – переходы этого автомата. При выполнении перехода заданное воздействие передается на тестируемую реализацию, после чего регистрируются реакции реализации и автоматически выносится вердикт о соответствии наблюдаемого поведения спецификации. В UniTESK алгоритм обхода конечного автомата реализован как внутренний компонент и не зависит от протокола и тестируемой системы.

Мутационные методы тестирования используются для обнаружения нестандартного поведения тестируемой системы (завершение из-за фатальной ошибки, «подвисание», ошибки доступа к памяти). Как правило, подобные ситуации не рассматриваются в спецификациях. В сообщения, сформированные на основе разработанной модели протокола, вносятся какие-либо изменения. Модель протокола позволяет изменять данные на любом этапе обмена, что дает возможность тестовому сценарию проходить через все значимые состояния протокола и в каждом таком состоянии проводить тестирование реализации в соответствии с

заданной программой.

3. ТЕСТОВЫЙ СТЕНД

Для тестирования реализаций клиента протокола TLS были использованы два сетевых узла. На одном узле функционирует модельная реализация под управлением UniTESK, выполняются основной поток управления тестовыми сценариями, обход тестового автомата и верификация наблюдаемых реакций. На другом узле функционирует тестируемая реализация. Тестовые сообщения протокола, сформированные модельной реализацией, передаются тестируемой системе, после чего регистрируются реакции тестируемого узла.

Работа с реализациями клиентов имеет свои особенности по сравнению с серверами. Сервер представляет собой непрерывно работающий процесс, прекращение работы этого процесса, как правило, является сбоем его работы. Клиентское приложение с точки зрения установления соединения работает короткое время и завершает свою работу либо установив соединение, либо ошибкой. Для многократного повторения этих действий требуется дополнительный агент.

В качестве реализаций клиента TLSv1.3 были выбраны:

– реализация TLS в виртуальной машине Java, JDK-14 (Java Development Kit) [8],

– реализация TLS библиотеки openssl-3.0.5 [9],

– интернет-браузер Mozilla Firefox, Portable Edition 97.0.1 [10],

– интернет-браузер Chromium, Portable Edition 103.0.5060.114 (Official Build, ungoogled-chromium) [11],

– интернет-браузер Atom 25.0.0.24 [12].

Первые две реализации являются частью широко используемых библиотек с открытым исходным кодом.

4. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

На данный момент в рамках технологии UniTESK (с использованием инструмента JavaTesK [13]) получены следующие результаты:

– расширена модель основной функциональности протокола TLS версии 1.3 для работы с реализациями клиента,

– разработан набор тестов для тестирования реализаций клиентов протокола, покрывающий часть требований спецификации.

Найдено несколько отклонений реализаций от спецификации.

JDK-14:

- В сообщениях ClientHello/ServerHello значения поля session_ID должны совпадать (для TLSv1.3), клиент должен это проверять. Реализация игнорирует данное поле.
- В сообщении ServerHello расширение signature_algorithms_cert с некорректными значениями игнорируется. Данное расширение задает допустимые алгоритмы подписи для сертификатов.
- Неизвестные расширения (а также дубликаты таких расширений) в сообщениях ServerHello/ServerHelloRetry игнорируются. Сообщение ServerHello может содержать только расширения, которые присутствовали в ClientHello (исключением является расширение Cookie). При этом расширения должны присутствовать в единственном экземпляре.
- Если сообщение ClientHello содержит недостаточно параметров, сервер может отправить сообщение ServerHelloRetry, предлагающее клиенту прислать исправленное сообщение ClientHello. Повторные сообщения ServerHelloRetry не допускаются. Реализация отвечает обычным образом на повторные сообщения ServerHelloRetry от сервера.
- Сообщение ServerHelloRetry должно содержать несколько обязательных расширений. Реализация клиента принимает сообщение ServerHelloRetry с единственным расширением Supported_versions (расширение необходимо, чтобы использовать версию TLS 1.3).
- Сообщение ServerHelloRetry в расширении KeyShare содержит алгоритм, который сервер желает использовать и который поддерживается клиентом (присутствует в ClientHello), но ключ для которого клиент не прислал. ServerHelloRetry не должно предлагать алгоритмы, ключи для которых уже присутствуют в ClientHello. Реализация клиента принимает такие некорректные сообщения.
- В ответ на ServerHelloRetry клиент присылает исправленное сообще-

ние ClientHello. Сервер отвечает сообщением ServerHello. При этом клиент должен проверить, что выбранный для сессии криптографический набор (поле CipherSuite) в ServerHelloRetry и ServerHello один и тот же. Реализация клиента не проверяет это требование.

Openssl-3.0.5:

- В сообщении ServerHello расширение signature_algorithms_cert с некорректными значениями игнорируется. Данное расширение задает допустимые алгоритмы подписи для сертификатов.
- В сообщении ServerHello игнорируется значение поля ProtocolVersion (версия протокола), если оно больше 3.3 (например, 4.2). Значение 3.3 соответствует последней на данный момент версии TLSv1.3.
- Неизвестные расширения (а также их дубликаты) в сообщениях ServerHello/ServerHelloRetry игнорируются. Сообщение ServerHello может содержать только расширения, которые присутствовали в ClientHello (исключением является расширение Cookie). При этом расширения должны присутствовать в единственном экземпляре.

Firefox 97.0.1:

Отклонений от спецификации не обнаружено.

Chromium 103.0.5060.114:

- Реализация не отвечает на сообщение KeyUpdate (с флагом 1), требующее обновить ключевой материал текущей сессии. Данное сообщение с указанным флагом требует от партнера подтвердить ответным сообщением новый ключевой материал.
- В сообщении CertificateRequest игнорируется наличие дополнительных и недопустимых расширений (кроме необходимого SignatureAndHash_Algorithm).

Atom 25.0.0.24:

- Реализация не отвечает на сообщение KeyUpdate (с флагом 1), требующее обновить ключевой материал текущей сессии. Данное сообщение с указанным флагом требует от партнера подтвердить ответным сообщением новый ключевой материал.
- В сообщении CertificateRequest игнорируется наличие дополнительных и недопустимых расширений (кроме необходимого SignatureAndHash_Algorithm).

ЗАКЛЮЧЕНИЕ

В работе представлен опыт верификации реализаций клиента криптографического протокола TLS версии 1.3.

TLS сегодня является одним из наиболее востребованных криптографических протоколов, предназначенных для создания защищенных каналов передачи данных. Протокол обеспечивает необходимую для своих задач функциональность: конфиденциальность передаваемых данных, целостность данных, аутентификацию сторон. В новой версии протокола TLS 1.3 была существенно переработана архитектура, устранен ряд недостатков предыдущих версий, выявленных как при разработке реализаций, так и в процессе их эксплуатации.

Нами был использован новый тестовый набор для верификации реализаций клиента протокола TLS 1.3 на соответствие спецификациям интернета, разработанный на основе спецификации RFC 8446 с использованием технологии UniTESK и методов мутационного тестирования. Для тестирования реализаций на соответствие формальным спецификациям применена технология UniTESK, предоставляющая средства автоматизации тестирования на основе использования конечных автоматов. Состояния тестируемой системы задают состояния автомата, а тестовые воздействия – переходы этого автомата. При выполнении перехода заданное воздействие передается на тестируемую реализацию, после чего регистрируются реакции реализации и автоматически выносится вердикт о соответствии наблюдаемого поведения спецификации. Мутационные методы тестирования использованы для обнаружения нестандартного поведения тестируемой системы (завершение из-за фатальной ошибки, «подвисание», ошибки доступа к памяти) с помощью передачи некорректных данных, такие ситуации часто остаются за рамками требований спецификаций. В сообщения, сформированные на основе разработанной модели протокола, вносятся какие-либо изменения. Модель протокола позволяет вносить изменения в поток данных на любом этапе сетевого обмена, что дает возможность тестовому сценарию проходить через все значимые состояния протокола и в каждом таком состоянии проводить тестирование реализации в соответствии с заданной программой.

Представленный подход доказал свою эффективность в наших предыдущих

проектах при тестировании сетевых протоколов, обеспечив обнаружение различных отклонений от спецификации и других ошибок [14–16]. Текущая работа является продолжением нашего проекта верификации протокола TLS 1.3 и охватывает реализации клиентской части протокола.

На данный момент обнаружено несколько отклонений реализаций JDK-14, openssl-3.0.5 и браузеров Chromium и Atom от спецификации.

Благодарности

Проект выполняется при поддержке РФФИ, проект № 20-07-00493 «Верификация функций безопасности и оценка устойчивости к атакам реализаций протокола TLS версии 1.3».

СПИСОК ЛИТЕРАТУРЫ

1. *Dierks T., Rescorla E.* The Transport Layer Security (TLS) Protocol Version 1.2. August 2008. IETF RFC 5246. URL: https://tools.ietf.org/html/rfc5246

2. *Rescorla E.* The Transport Layer Security (TLS) Protocol Version 1.3. August 2018. IETF RFC 8446. URL: https://tools.ietf.org/html/rfc8446

3. *Никешин А.В., Шнитман В.З.* Верификация функций безопасности протокола TLS версии 1.3 // Научный сервис в сети Интернет: труды XXII Всероссийской научной конференции (21–25 сентября 2020 г., онлайн). М.: ИПМ им. М.В. Келдыша, 2020. С. 515–526. https://doi.org/10.20948/abrau-2020-22

4. *Никешин А.В., Шнитман В.З.* Верификация функций безопасности расширений протокола TLS 1.3 // Научный сервис в сети Интернет: труды XXIII Всероссийской научной конференции (20–23 сентября 2021 г., онлайн). М.: ИПМ им. М.В. Келдыша, 2021. С. 251–264. https://doi.org/10.20948/abrau-2021-14

5. *Eastlake D.* Transport Layer Security (TLS) Extensions: Extension Definitions. January 2011. IETF RFC 6066. URL: https://tools.ietf.org/html/rfc6066

6. *Thomson M.* Record Size Limit Extension for TLS. August 2018. IETF RFC 8449. URL: https://tools.ietf.org/html/rfc8449

7. *Bourdonov I., Kossatchev A., Kuliamin V., Petrenko A.* UniTesK Test Suite Architecture // Proceedings of FME 2002. LNCS 2391. P. 77–88, Springer-Verlag, 2002.

8. *Java Development Kit 14.0.1 GA.* URL: https://jdk.java.net/14/

9. *OpenSSL Project.* URL: https://www.openssl.org/

10. *Mozilla Firefox, Portable Edition 97.0.1.*
URL: https://portableapps.com/apps/internet/firefox_portable/

11. *Ungoogled Chromium Portable 103.0.5060.114.*
URL: https://portapps.io/app/ungoogled-chromium-portable/

12. *Atom 25.0.0.24.* URL: https://browser.ru/

13. *JavaTESK.* URL: http://www.unitesk.ru/content/category/5/25/60/

14. *Никешин А.В., Пакулин Н.В., Шнитман В.З.* Разработка тестового набора для верификации реализаций протокола безопасности TLS // Труды ИСП РАН. 2012. Том 23. С. 387–404.

15. *Никешин А.В., Пакулин Н.В., Шнитман В.З.* Тестирование реализаций клиента протокола TLS // Труды ИСП РАН. 2015. Т. 27, вып. 2. С. 145–160.

16. *Никешин А.В., Шнитман В.З.* Тестирование соответствия реализаций протокола EAP и его методов спецификациям Интернета // Труды ИСП РАН. 2018. Т. 30, вып. 6. С. 89–104. https://doi.org/10.15514/ISPRAS-2018-30(6)-5

EXPERIENCE OF TLS 1.3 CLIENTS VERIFICATION

A. V. Nikeshin[1] [0000-0001-5781-9736], V. Z. Shnitman[2] [0000-0002-1509-0972]

Ivannikov Institute for System Programming of the Russian Academy of Sciences, Alexander Solzhenitsyn st., 25, Moscow, 109004

[1]alexn@ispras.ru, [2]vzs@ispras.ru

Abstract

This paper presents the experience of verifying client implementations of the TLS cryptographic protocol version 1.3. TLS is a widely used cryptographic protocol today, designed to create secure data transmission channels. The protocol provides the necessary functionality for its tasks: confidentiality of transmitted data, data integrity, and authentication of the parties. In the new version 1.3 of the TLS architecture was significantly redesigned, eliminating a number of shortcomings of previous versions that were identified both during the development of implementations and during their operation. We used a new test suite for verifying client implementations of the TLS 1.3 for compliance with Internet specifications, developed on the basis of the RFC8446,

using UniTESK technology and mutation testing methods. To test implementations for compliance with formal specifications, UniTESK technology is used, which provides testing automation tools based on the use of finite state machines. The states of the system under test define the states of the state machine, and the test effects are the transitions of this machine. When performing a transition, the specified impact is passed to the implementation under test, after which the implementation's reactions are recorded and a verdict is automatically made on the compliance of the observed behavior with the specification. Mutational testing methods are used to detect non-standard behavior of the system under test by transmitting incorrect data. Some changes are made to the protocol exchange flow created in accordance with the specification: either the values of the message fields formed on the basis of the developed protocol model are changed, or the order of messages in the exchange flow is changed. The protocol model allows one to make changes to the data flow at any stage of the network exchange, which allows the test scenario to pass through all the significant states of the protocol and in each such state to test the implementation in accordance with the specified program. The presented approach has proven effective in several of our projects when testing network protocols, providing detection of various deviations from the specification and other errors. The current work is part of the TLS 1.3 protocol verification project and covers TLS client implementations.

Keywords*: security, TLS, TLSv1.3, protocols, testing, verification, evaluate robustness, Internet, standards, formal specifications.*

REFERENCES

1. *Dierks T., Rescorla E.* The Transport Layer Security (TLS) Protocol Version 1.2. August 2008. IETF RFC 5246. URL: https://tools.ietf.org/html/rfc5246

2. *Rescorla E.* The Transport Layer Security (TLS) Protocol Version 1.3. August 2018. IETF RFC 8446. URL: https://tools.ietf.org/html/rfc8446

3. *Nikeshin A.V., Shnitman V.Z.* Verification of security properties of the TLS protocol version 1.3 // Nauchnyi servis v seti Internet: trudy XXII Vserossiiskoi nauchnoi konferentsii (21–25 sentiabria 2020 g., online). M.: IPM im. M.V. Keldysha, 2020. P. 515–526. https://doi.org/10.20948/abrau-2020-22

4. *Nikeshin A.V., Shnitman V.Z.* Verification of security properties of the TLS 1.3

extensions // Nauchnyi servis v seti Internet: trudy XXIII Vserossiiskoi nauchnoi konferentsii (20–23 sentiabria 2021 g., online). M.: IPM im. M.V. Keldysha, 2021. P. 251–264. https://doi.org/10.20948/abrau-2021-14

5. *Eastlake D.* Transport Layer Security (TLS) Extensions: Extension Definitions. January 2011. IETF RFC 6066. URL: https://tools.ietf.org/html/rfc6066

6. *Thomson M.* Record Size Limit Extension for TLS. August 2018. IETF RFC 8449. URL: https://tools.ietf.org/html/rfc8449

7. *Bourdonov I., Kossatchev A., Kuliamin V., Petrenko A.* UniTesK Test Suite Architecture // Proceedings of FME 2002. LNCS 2391. P. 77–88, Springer-Verlag, 2002.

8. *Java Development Kit 14.0.1 GA.* URL: https://jdk.java.net/14/

9. *OpenSSL Project.* URL: https://www.openssl.org/

10. *Mozilla Firefox, Portable Edition 97.0.1.*
URL: https://portableapps.com/apps/internet/firefox_portable/

11. *Ungoogled Chromium Portable 103.0.5060.114.*
URL: https://portapps.io/app/ungoogled-chromium-portable/

12. *Atom 25.0.0.24.* URL: https://browser.ru/

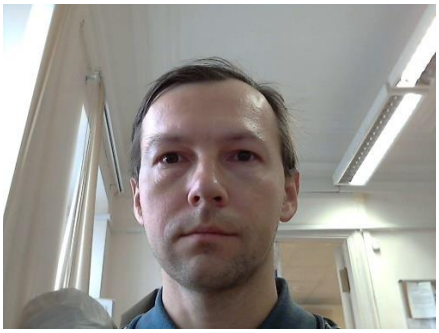
13. *JavaTESK.* URL: http://www.unitesk.ru/content/category/5/25/60/

14. *Nikeshin A.V., Pakulin N.V., Shnitman V.Z.* Razrabotka testovogo nabora dlya verifikatsii realizatsiy protokola bezopasnosti TLS // Trudy ISP RAN /Proc. ISP RAS. 2012. Vol. 23. P. 387–404.

15. *Nikeshin A.V., Pakulin N.V., Shnitman V.Z.* TLS Clients Testing // Trudy ISP RAN /Proc. ISP RAS. 2015. Vol. 27, issue 2. P. 145–160.

16. *Nikeshin A.V., Shnitman V.Z.* Conformance testing of Extensible Authentication Protocol implementations // Trudy ISP RAN/Proc. ISP RAS. 2018. Vol. 30, issue 6. P. 89–104 (in Russian). https://doi.org/10.15514/ISPRAS-2018-30(6)-5

СВЕДЕНИЯ ОБ АВТОРАХ



НИКЕШИН Алексей Вячеславович – научный сотрудник Института системного программирования им. В.П. Иванникова РАН.

Aleksey Vyacheslavovich NIKESHIN – researcher, Ivannikov Institute for System Programming of the RAS.

email: alexn@ispras.ru;
ORCID: 0000-0001-5781-9736

ШНИТМАН Виктор Зиновьевич – д. т. н., с. н. с., заведующий отделом Института системного программирования им. В.П. Иванникова РАН.

Victor Zinovievich SHNITMAN – Doctor of Technical Sciences, Senior Research Officer, Head of Department, Ivannikov Institute for System Programming of the RAS.

email: vzs@ispras.ru;
ORCID: 0000-0002-1509-0972

Материал поступил в редакцию 30 января 2023 года

УДК 519.6; 519.2

СТАТИСТИЧЕСКИЙ АНАЛИЗ ДАННЫХ НАБЛЮДЕНИЙ ПОТОКОВ ВЗАИМОДЕЙСТВИЯ ОКЕАНА И АТМОСФЕРЫ В СЕВЕРНОЙ АТЛАНТИКЕ

Н. П. Тучкова[1] [0000-0001-5357-9640], К. П. Беляев[2] [0000-0003-2111-2709], Г. М. Михайлов[3] [0000-0002-4535-7180]

[1, 2, 3]Вычислительный центр им. А.А. Дородницына ФИЦ Информатика и управление РАН, г. Москва

[2]Институт океанологии им. П.П. Ширшова РАН, г. Москва

[1]natalia_tuchkova@mail.ru, [2]kosbel55@gmail.com, [3]gmickail@ccas.ru,

Аннотация

Проанализированы данные наблюдений 1979–2018 гг. в районе Северной Атлантики, полученные в результате реализации проекта Российской академии наук по исследованию атмосферы в Северной Атлантике (РАН-НААД). Набор данных предоставляет множество параметров поверхности и свободной атмосферы на основе сигма-модели и отвечает многим требованиям метеорологов, климатологов и океанографов, работающих как в исследовательской, так и в оперативной областях. Проведен анализ сезонной и многолетней изменчивости тепловых потоков и температуры поверхности воды в Северной Атлантике. В качестве основного метода исследования использованы схемы анализа диффузионных процессов. На основе заданных рядов длиной в 40 лет с 1979 по 2018 годы вычислены такие параметры диффузионных процессов, как среднее (снос процесса) и дисперсия (диффузия процесса) и построены их карты и временные кривые. Численные расчеты выполнены на суперкомпьютере Ломоносов-2 Московского государственного университета имени М.В. Ломоносова.

Ключевые слова*: анализ временных рядов, климатический сезонный ход, максимальные и минимальные значения давления внутри климатического года*

ВВЕДЕНИЕ

Работа посвящена вероятностному анализу данных наблюдений в районе Северной Атлантики. Предварительные исследования, проводимые в рамках

проекта РАН-НААД [1], позволили получить 40-летний трехмерный ретроспективный прогноз атмосферы Северной Атлантики (10°–80° с. ш.) с пространственным разрешением 14 км и 50-ю уровнями в вертикальном направлении (до 50 гПа). Прогноз выполнен с региональной настройкой модели WRF-ARW3.8.1[1] для периода 1979–2018 гг. и значений реанализа ERA-Interim[2] в качестве граничных условий. Набор данных предоставляет множество параметров поверхности и свободной атмосферы на основе сигма-модели в границах выбранного региона.

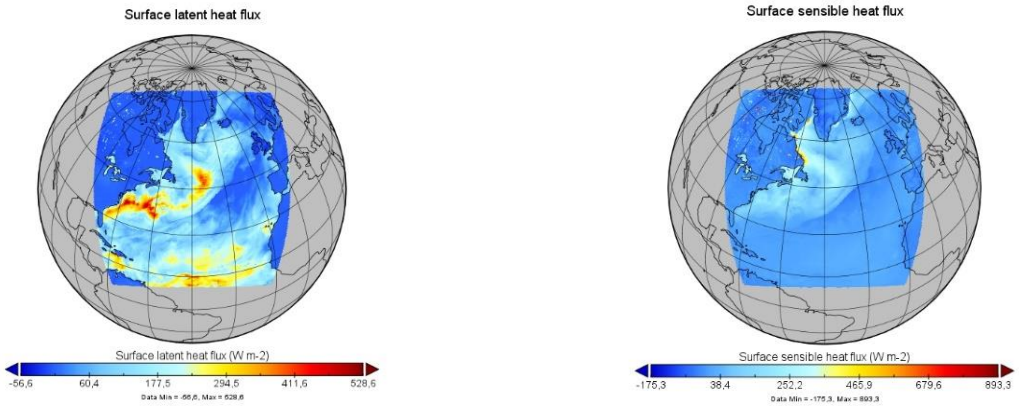
Предпосылки этих исследований определяются востребованностью развития методов математического моделирования и прогнозирования в области экологии. Приложение результатов исследований состоит в развитии методов вероятностного анализа для оценки физических характеристик диффузионных процессов на примере полей температуры и потоков тепла, а также оценки начальных значений при численном моделировании.

Проанализированы такие параметры, как приводная температура воздуха (2 m temperature, °C) и тепловые потоки океан-атмосфера, а именно, поток явного тепла (surface sensible heat flux Wm^{-2}, $Вт/m^{-2}$), поток скрытого тепла (surface latent heat flux Wm^{-2}, $Вт/m^{2}$). Массив данных составляют значения за 40 лет в каждой точке сетки (south_north=550, west_east=550) с интервалом измерений в 3 часа (т. е. 2920 измерений в год для невисокосных и 2928 для високосных лет, соответственно). На рис. 1 (а, б) приведены примеры значений наблюдаемых величин явного и скрытого тепла на 1 января 2015 г. в Северной Атлантике.

Метод, предложенный в статье, достаточно известен в теории временных рядов [2], однако для анализа потоков тепла ранее не применялся. Нами показано, что этим методом можно выявить новые интересные закономерности. Ранее авторы проводили аналогичные исследования с одним параметром, полем давления [3].

[1] https://www.mmm.ucar.edu/weather-research-and-forecasting-model

[2] https://www.ecmwf.int/en/forecasts/datasets/reanalysis-datasets/era-interim



(а) (б)

Рис. 1. Значения потоков тепла (Wm^{-2}) 01.01.2015: а) явного; б) скрытого

На начальном этапе данного исследования были получены предварительные оценки для наблюдаемых в регионе значений суммарного тепла и температуры воздуха (температура на высоте 2 м над уровнем поверхности воды). На рис. 2 показан пример наблюдаемого поля температуры в выбранном регионе на 1 января 2015 г.

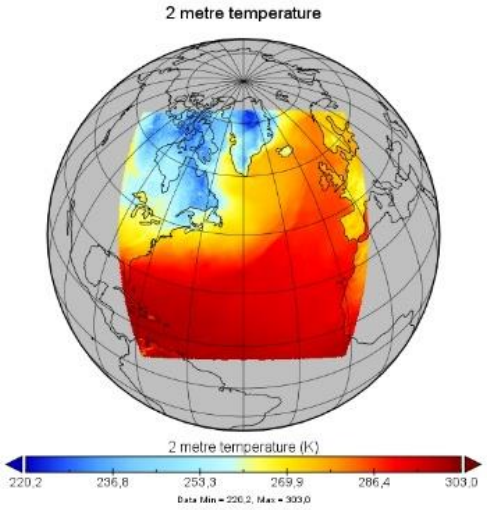


Рис. 2. Значения поля температуры (K) в регионе 01.01.2015

На рис. 3–5 представлены графики сезонного хода минимумов, максимумов и частотного распределения температуры и потоков суммарного тепла (явного и скрытого), полученные из анализа данных эксперимента РАН-НААД.

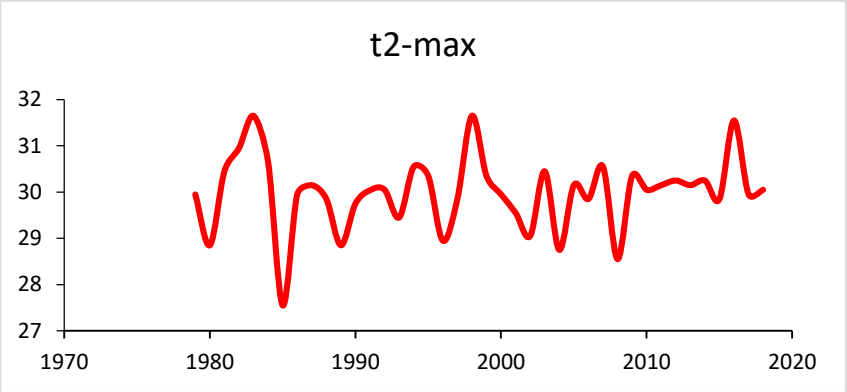


Рис. 3. Сезонный ход максимумов поверхностной температуры по всему региону с 1979 по 2018 гг.

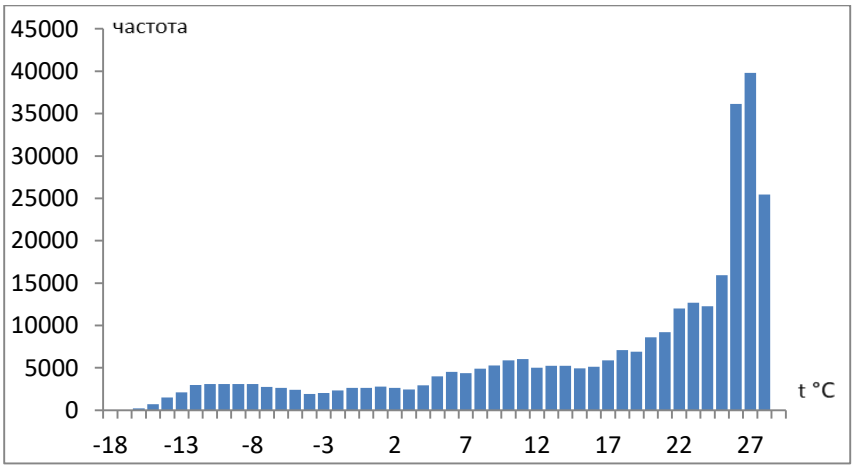


Рис. 4. Гистограмма частот средней температуры по всему региону за январь и весь период наблюдений с 1979 по 2018 гг.

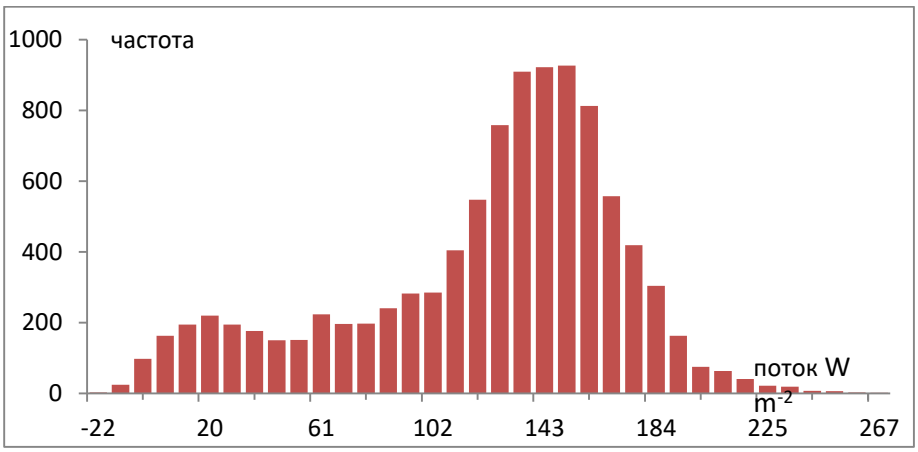


Рис. 5. Гистограмма частот среднего суммарного потока тепла, усредненного по всему региону на 1 января за весь период наблюдений с 1979 по 2018 гг.

Аналогично предыдущим исследованиям авторов [3], изменчивость случайного процесса представляется в виде

$$dX = a(t, X)dt + b(t, x)dW, \quad (1)$$

где X – значение поля (температуры и потока тепла, в данном случае применительно к настоящим исследованиям) в момент времени *t* в точке с заданными координатами, *t* – время, *dW* – стандартное обозначение гауссова «белого шума» – обобщенного случайного процесса с нулевым средним значением и дисперсией, равной единице, при этом его ковариационная функция равна дельта-функции, то есть $EdW(t)dW(\tau) = \delta(t-\tau)$. Здесь и далее $\delta(t-\tau)$=1, если $t = \tau$, и нулю, если нет, $a(t, X), b(t, x)$ – некоторые функции. Выражение (1) понимается в интегральном смысле, то есть

$$X(t+\Delta t) - X(t) = \int_t^{t+\Delta t} a(u, X)du + \int_t^{t+\Delta t} b(u, X)[W(u+du) - W(u)]. \quad (2)$$

В формуле (2) выражение $W(u+du) - W(u)$ представляет собой гауссову случайную величину с нулевым средним и дисперсией, равной *du*. Теория стохастического интеграла и все определения, необходимые для понимания формул (1) и (2), содержатся в [4–6].

Согласно первоисточнику [4], для определения коэффициентов $a(t, X)$ и $b(t, x)$ применяют следующие выражения:

$$a(t, x) = (dt)^{-1} \int_t^{t+dt} (y - x)\, p(y \mid x)dy, \quad (3)$$

$$b^2(t, x) = (dt)^{-1} \int_t^{t+dt} (y - x)^2\, p(y \mid x)dy, \quad (4)$$

где в формулах (3) и (4) использованы следующие обозначения:

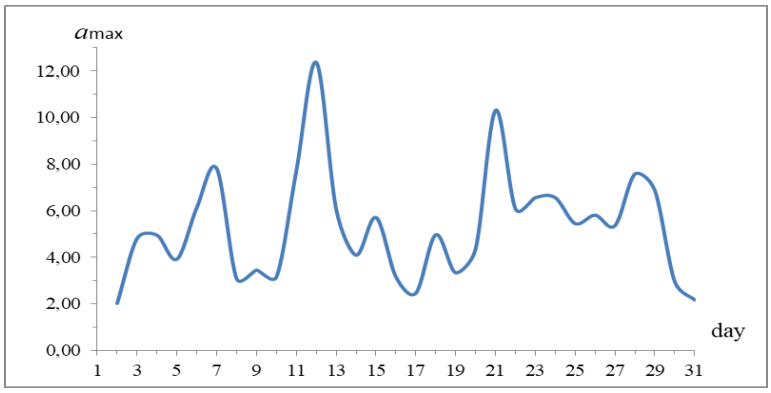
y, x – значения процесса *X(t)* в моменты времени *t+dt* и *t;*

p(y|x)dt – вероятность (условная вероятность) события, когда значения *X(t+dt)=y* при условии *X(t)=x*, то есть когда выполняется равенство

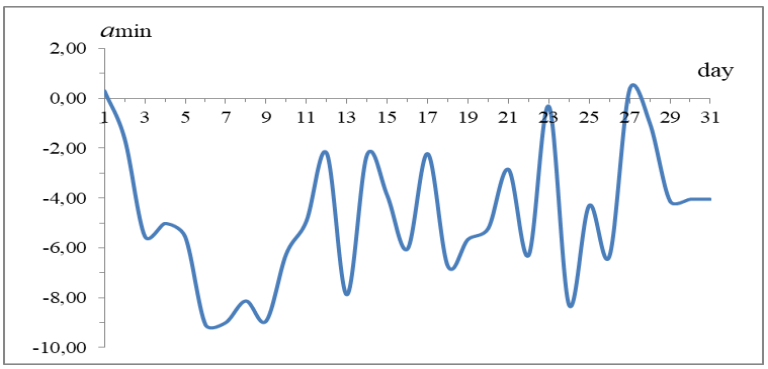
$$p(y \mid x)dt = P(X(t+dt) = y \mid P(t) = x).$$

Ставится задача – вычислить эти коэффициенты и произвести анализ полученных характеристик.

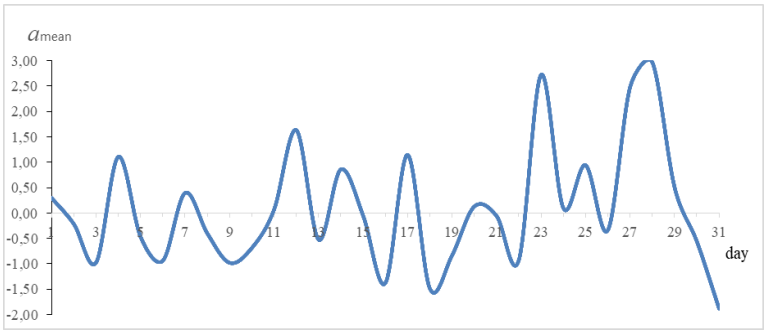
По приведенным формулам были произведены численные расчеты. Далее были проанализированы их результаты, некоторые из них, а именно, статистические характеристики параметров $a(t, X)$ и $b^2(t, x)$ за климатический январь, представлены на рис. 6–9. Такие характеристики получены на каждый климатический месяц за весь период наблюдений и на всем пространстве измерений. Из этих оценок следует, что среднее значение климатического месяца соответствует годичному циклу климатического года.



(а)



(б)



(в)

Рис. 6. Кривые поведения коэффициента $a(t, X)$ для климатического января суммарного потока: (а) максимумы;(б) минимумы; (в) средние по всему региону за 1979–2018 гг.

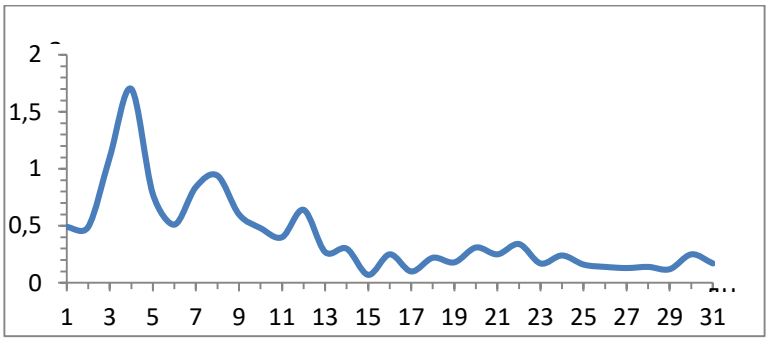


Рис. 7. Кривая значений максимумов коэффициента $a(t, X)$ для значений температуры усредненного января за 1979–2018 гг. по всему региону

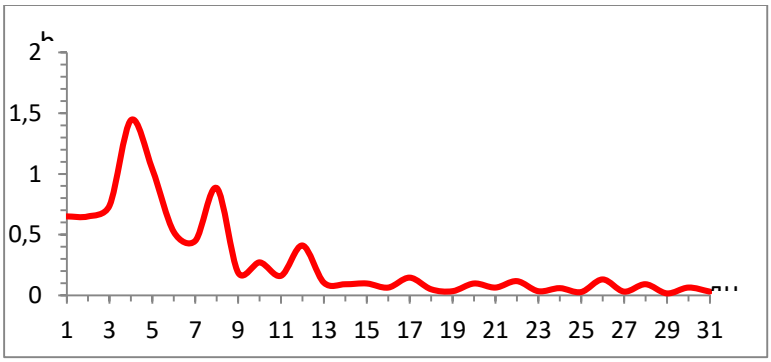


Рис. 8. Кривая значений максимумов коэффициента $b^2(t, x)$ для значений температуры для усредненного января за 1979–2018 гг. по всему региону

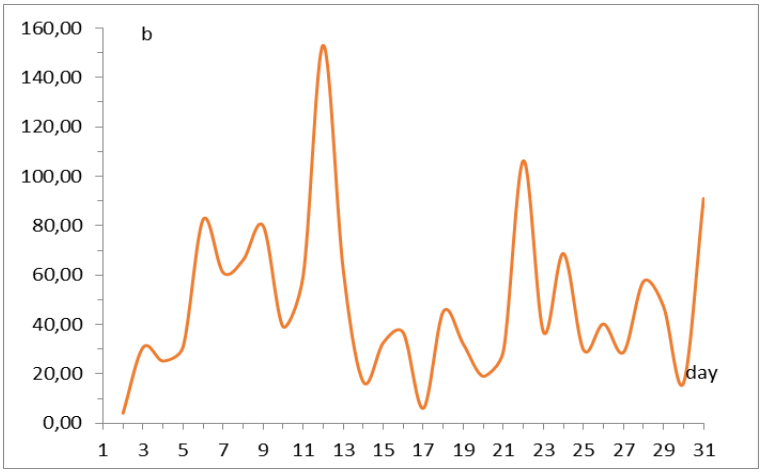


Рис. 9. Кривая значений коэффициента $b^2(t,x)$ для значений суммарного потока климатического *декабря* за 1979–2018 гг. по *всему* региону

Из приведенных иллюстраций видно, что для климатического декабря достаточно хорошо выражена синоптическая изменчивость, период которой составляет приблизительно 10 суток. Этот результат не противоречит известной синоптической изменчивости потоков тепла, определяемой циклонической активностью, которая составляет примерно 3–5 суток. Коэффициенты $a(t,X)$ и $b^2(t,x)$ отражают последовательную изменчивость двух дней подряд, что определяет период около 10 суток.

В исследовании были также получены аппроксимирующие функции *A*(*t*,*X*) и *B*(*t*,*x*) для значений климатического года за весь период исследований и на всем пространстве Северной Атлантики (10°–80° с. ш.) с пространственным разрешением 14 км. Эти функции позволяют получить, как результат, численную оценку изменчивости случайного процесса.

ЗАКЛЮЧЕНИЕ И ВЫВОДЫ

В работе показано, что в многолетней изменчивости максимумов (за 40 лет) по акватории приводной температуры воздуха заметно присутствует 11-летний цикл, обусловленный известными в природе циклами солнечной активности. При этом не выявлено тенденций к росту этих максимумов за период 40 лет. Для вычисленных коэффициентов средней диффузии за климатический январь выявлены также синоптические колебания порядка 3–5 дней, которые хорошо видны

на графиках. Такие закономерности ранее не были известны и выявлены с помощью нового метода исследования на основе анализа больших данных, массивы которых формируются из наблюдений и реанализа в рамках исследований мирового океана. Результаты могут быть использованы для аналитического исследования многолетнего поведения изучаемых процессов.

БЛАГОДАРНОСТИ

Работа представлена в рамках выполнения темы НИР 0063-2019-0003 ФИЦ ИУ РАН и темы НИР 0128-2021-0002 ИО РАН.

СПИСОК ЛИТЕРАТУРЫ

1. *Gavrikov A., Gulev S., Markina M., Tilinina N., Verezemskaya P., Barnier B., Dufour A., Zolina O., Zyulyaeva Y., Krinitskiy M., Okhlopkov I., Sokov A.* RAS-NAAD: 40-yr High-Resolution North Atlantic Atmospheric Hindcast for Multipurpose Applications (New Dataset for the Regional Mesoscale Studies in the Atmosphere and the Ocean) // Journal of Applied Meteorology and Climatology. 2020. V. 59, issue 5. P. 793–817. https://doi.org/10.1175/JAMC-D-19-0190.1.

2. *Kendall M., Stuart A., Ord J.K.* The Advanced Theory of Statistics. Volume 3: Design and Analysis, and Time-Series. Fourth edition Hardcover – March 13, 1983.

3. *Belyaev K., Mikhaylov G., Salnikov A., Tuchkova N.* Seasonal and Decadal Variability of Atmosphere Pressure in Arctic, its Statistical and Temporal Analysis // CEUR Workshop Proceedings, M. Jeusfeld c/o Redaktion Sun SITE, Informatik V, RWTH Aachen (Aachen, Germany). 2020. V. 2784. P. 51-61.
URL: http://ceur-ws.org/Vol-2784/rpaper05.pdf.

4. *Гихман И., Скороход А.* Введение в теорию случайных процессов. М.: Наука, 1965. 655 с.

5. *Назаров А., Терпунов А.* Теория вероятностей и случайных процессов. Изд-во Томского госуниверситета, 2010. 204 с.

6. *Risken H.* The Fokker–Planck Equation: Methods of Solutions and Applications. Springer. 1984. 452 p.

STATISTICAL ANALYSIS OF OBSERVATION DATA OF AIR-SEA INTERACTION IN THE NORTH ATLANTIC

N. P. Tuchkova[1] [0000-0001-5357-9640], K. P. Belyaev[2] [0000-0003-2111-2709],
G. M. Mikhaylov[3] [0000-0002-4535-7180]

[1, 2, 3]Dorodnicyn Computing Center FRC CSC of RAS, Vavilov str., 40, 11933, Moscow
[2]Shirshov Institute of Oceanology of RAS, Nahimovskiy pr., 36, 117218, Moscow

[1]natalia_tuchkova@mail.ru, [2]kosbel55@gmail.com, [3]gmickail@ccas.ru

Abstract

The observational data for 1979-2018 in the North Atlantic region are analyzed. These data were obtained as a result of the implementation of the project of the Russian Academy of Sciences for the study of the atmosphere in the North Atlantic (RAS-NAAD). The dataset provides many surface and free atmosphere parameters based on the sigma model and meets the many requirements of meteorologists, climatologists and oceanographers working in both research and operational fields. The paper analyzes the seasonal and long-term variability of the field of heat fluxes and water surface temperature in the North Atlantic. Schemes for analyzing diffusion processes were used as the main research method. Based on the given series of 40 years in length from 1979 to 2018, such parameters of diffusion processes as the mean (process drift) and variance (process diffusion) were calculated and their maps and time curves were constructed. Numerical calculations realized on the Lomonosov-2 supercomputer of the Lomonosov Moscow State University.

***Keywords**:* time series analysis, climatic seasonal cycle, maximum and minimum heat fluxes and temperature values within a climatic year.

REFERENCES

1. *Gavrikov A., Gulev S., Markina M., Tilinina N., Verezemskaya P., Barnier B., Dufour A., Zolina O., Zyulyaeva Y., Krinitskiy M., Okhlopkov I., Sokov A*. RAS-NAAD: 40-yr High-Resolution North Atlantic Atmospheric Hindcast for Multipurpose Applications (New Dataset for the Regional Mesoscale Studies in the Atmosphere and the Ocean) // Journal of Applied Meteorology and Climatology. 2020, V. 59, issue 5. P. 793–817. https://doi.org/10.1175/JAMC-D-19-0190.1.

2. *Kendall M., Stuart A., Ord J.K.* The Advanced Theory of Statistics. V. 3: Design and Analysis, and Time-Series. Fourth edition Hardcover – March 13, 1983.

3. *Belyaev K., Mikhaylov G., Salnikov A., Tuchkova N.* Seasonal and Decadal Variability of Atmosphere Pressure in Arctic, its Statistical and Temporal Analysis // CEUR Workshop Proceedings, M. Jeusfeld c/o Redaktion Sun SITE, Informatik V, RWTH Aachen (Aachen, Germany), 2020. V. 2784. P. 51-61.
URL: http://ceur-ws.org/Vol-2784/rpaper05.pdf.

4. *Gihman I., Skorohod A.* Vvedeniye v teoriyu sluchajnyh processov. M.: Nauka, 1965. 655 p.

5. *Nazarov A., Terpunov A.* Teoriya veroyatnostej i sluchajnyh processov. Izd-vo Tomskogo Gosuniversiteta. 2010. 204 p.

6. *Risken H.* The Fokker–Planck Equation: Methods of Solutions and Applications. Springer. 1984. 452 p.

СВЕДЕНИЯ ОБ АВТОРАХ

ТУЧКОВА Наталия Павловна – старший научный сотрудник Вычислительного центра им. А.А. Дородницына ФИЦ ИУ РАН, кандидат физ.-мат. наук, окончила ВМиК МГУ им. М.В. Ломоносова. Специалист в области алгоритмических языков и информационных технологий.

Natalia Pavlovna TUCHKOVA – senior researcher of Dorodnicyn computing center FRC SCS RAS, PhD in physics with a math degree, graduated from CS Faculty of Lomonosov MSU. The expert in the field of algorithmic languages and information technologies.

email: natalia_tuchkova@mail.ru

ORCID: 0000-0001-5357-9640

БЕЛЯЕВ Константин Павлович – ведущий научный сотрудник Института океанологии им. П.П. Ширшова РАН и ФИЦ ИУ, доктор физ.-мат. наук, профессор кафедры теории вероятностей и статистики ВМиК МГУ им. М.В. Ломоносова. Сфера научных интересов – математическое моделирование и усвоение данных наблюдений, статистический анализ натурных данных.

Konstantin Pavlovich BELYAEV – leading scientist of Shirshov Institute of Oceanology, Russian Academy of Science. Doctor of science, professor of Dept. of Appiied Math and Cybernetics, Lomonosov Moscow State University. Research interests – math. modelling and data assimilation, statistical analysis of natural data.

email: kosbel55@gmail.com

ORCID: 0000-0003-2111-2709

МИХАЙЛОВ Гурий Михайлович – ведущий научный сотрудник Вычислительного центра им. А.А. Дородницына ФИЦ ИУ РАН, кандидат физ.-мат. наук. Сфера научных интересов – архитектура вычислительных систем и сетей, вычислительные и информационные технологии.

Gury Mickailovich Mikhaylov – leading scientist of Dorodnicyn computing center FRC SCS RAS, PhD in physics with a math degree. Research interests include architecture of computing systems and networks, computing and information technology.

email: gmickail@ccas.ru

ORCID: 0000-0002-4535-7180

Материал поступил в редакцию 30 января 2023 года