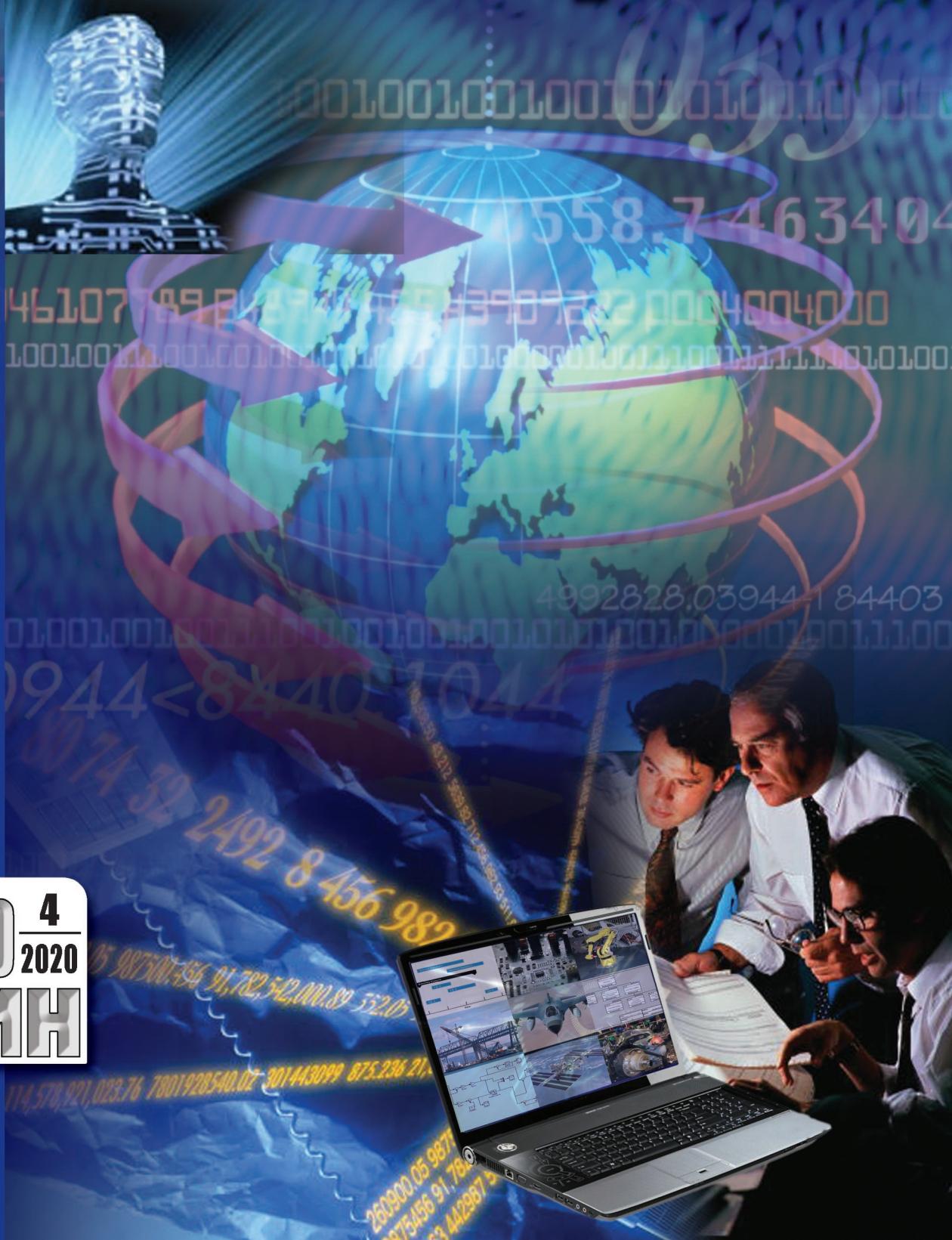


ISSN 2220-3397

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Программная инженерия



Пр
4
2020
Том 11
ИН



ЦЕНТР
ПРИКЛАДНЫХ
ИССЛЕДОВАНИЙ
КОМПЬЮТЕРНЫХ
СЕТЕЙ



**3-я Международная научно-техническая конференция
«Современные сетевые технологии»
«Modern Network Technologies, MoNeTec-2020»**

27 – 29 октября 2020 г., Москва

Конференция собирает представителей международного научного сообщества, исследовательских подразделений корпораций, стартапов, промышленности и бизнеса, институтов развития и органов государственной власти для обсуждения перспективных и актуальных технологий в сфере компьютерных сетей, виртуализации сетевых ресурсов и облачных вычислений, использования методов искусственного интеллекта.

Основные темы конференции:

- Архитектура и протоколы для автоматизации управления сетями и оркестрации в облачных инфраструктурах
- Инновационные технологии для облаков, Интернета, компьютерных сетей
- Новые парадигмы в организации и функционирования сетей, например, Intent Based Network, Information Centric Network, Content Centric Network и т.п.
- Подходы, методы и средства для управления качеством сервиса и распределения ресурсов в сетях и облачных средах
- Применение больших данных и машинного обучения для повышения эффективности функционирования сетей, облачных платформ и управления ими
- Проблемы безопасности сетей и облачных платформ
- Применение сетевых облачных технологий к обработке больших данных, Интернета вещей, сетей беспроводной и мобильной связи
- Вопросы архитектуры программно-управляемого телекоммуникационного оборудования
- Программно-конфигурируемые радиосистемы приема, передачи и обработки информации

Организаторы конференции – Консорциум «Современные сетевые технологии», Центр прикладных исследований компьютерных сетей.



Место проведения: Международный центр по информатике и электронике (ИнтерЭВМ), улица Пресненский Вал, д.19., Москва

Подробная информация о конференции размещена на официальном сайте
<http://www.monetc.ru>

Программная инженерия

Том 11
№ 4
2020
Прин

Учредитель: Издательство "НОВЫЕ ТЕХНОЛОГИИ"

Издается с сентября 2010 г.

DOI 10.17587/issn.2220-3397

ISSN 2220-3397

Редакционный совет

Садовничий В.А., акад. РАН
(председатель)
Бетелин В.Б., акад. РАН
Васильев В.Н., чл.-корр. РАН
Жижченко А.Б., акад. РАН
Макаров В.Л., акад. РАН
Панченко В.Я., акад. РАН
Стемпковский А.Л., акад. РАН
Ухлинов Л.М., д.т.н.
Федоров И.Б., акад. РАН
Четверушкин Б.Н., акад. РАН

Главный редактор

Васенин В.А., д.ф.-м.н., проф.

Редколлегия

Антонов Б.И.
Афонин С.А., к.ф.-м.н.
Бурдонов И.Б., д.ф.-м.н., проф.
Борзовс Ю., проф. (Латвия)
Гаврилов А.В., к.т.н.
Галатенко А.В., к.ф.-м.н.
Корнеев В.В., д.т.н., проф.
Костюхин К.А., к.ф.-м.н.
Махортов С.Д., д.ф.-м.н., доц.
Манцивода А.В., д.ф.-м.н., доц.
Назиров Р.Р., д.т.н., проф.
Нечаев В.В., д.т.н., проф.
Новиков Б.А., д.ф.-м.н., проф.
Павлов В.Л. (США)
Пальчунов Д.Е., д.ф.-м.н., доц.
Петренко А.К., д.ф.-м.н., проф.
Позднеев Б.М., д.т.н., проф.
Позин Б.А., д.т.н., проф.
Серебряков В.А., д.ф.-м.н., проф.
Сорокин А.В., к.т.н., доц.
Терехов А.Н., д.ф.-м.н., проф.
Филимонов Н.Б., д.т.н., проф.
Шапченко К.А., к.ф.-м.н.
Шундеев А.С., к.ф.-м.н.
Щур Л.Н., д.ф.-м.н., проф.
Язов Ю.К., д.т.н., проф.
Якобсон И., проф. (Швейцария)

Редакция

Лысенко А.В., Чугунова А.В.

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

СОДЕРЖАНИЕ

Сидняев Н. И., Бутенко Ю. И., Болотова Е. Е. Логическая модель требований информационно-системной надежности для баз знаний интеллектуальных систем	195
Булгаков Д. Ю., Булгакова Я. В., Карапыгин Н. А. Современное свободное программное обеспечение для анализа и обработки электроэнцефалограмм: возможности и выбор	205
Мальцев А. В. Методы распределенного рендеринга виртуальных сцен на GPU с реализацией эффекта размытия в движении	213
Орлова Е. В. Методы и модели анализа данных и машинного обучения в задаче управления производительностью труда	219
Девянин П. Н., Леонова М. А. Применение подтипов и тотальных функций формального метода Event-B для описания и верификации МРОСЛ ДП-модели	230
Черепнёв М. А. Оценка скорости работы нового параллельного блочного алгоритма для решения задач в области больших разреженных систем над большим простым полем	242
Носков С. И., Вергасов А. С. Регрессионная модель структурных факторов киберугроз	251

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

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

© Издательство "Новые технологии", "Программная инженерия", 2020

SOFTWARE ENGINEERING***PROGRAMMAYA INGENERIA***

Published since September 2010

DOI 10.17587/issn.2220-3397

ISSN 2220-3397

Editorial Council:

SADOVNICHY V. A., Dr. Sci. (Phys.-Math.), Acad. RAS (*Head*)
 BETELIN V. B., Dr. Sci. (Phys.-Math.), Acad. RAS
 VASIL'EV V. N., Dr. Sci. (Tech.), Cor.-Mem. RAS
 ZHIZHCHEVKO A. B., Dr. Sci. (Phys.-Math.), Acad. RAS
 MAKAROV V. L., Dr. Sci. (Phys.-Math.), Acad. RAS
 PANCHENKO V. YA., Dr. Sci. (Phys.-Math.), Acad. RAS
 STEMPKOVSKY A. L., Dr. Sci. (Tech.), Acad. RAS
 UKHLINOV L. M., Dr. Sci. (Tech.)
 FEDOROV I. B., Dr. Sci. (Tech.), Acad. RAS
 CHETVERTUSHKIN B. N., Dr. Sci. (Phys.-Math.), Acad. RAS

Editor-in-Chief:

VASENIN V. A., Dr. Sci. (Phys.-Math.)

Editorial Board:

ANTONOV B.I.
 AFONIN S.A., Cand. Sci. (Phys.-Math)
 BURDONOV I.B., Dr. Sci. (Phys.-Math)
 BORZOVS JURIS, Dr. Sci. (Comp. Sci), Latvia
 GALATENKO A.V., Cand. Sci. (Phys.-Math)
 GAVRILOV A.V., Cand. Sci. (Tech)
 JACOBSON IVAR, Dr. Sci. (Philos., Comp. Sci.), Switzerland
 KORNEEV V.V., Dr. Sci. (Tech)
 KOSTYUKHIN K.A., Cand. Sci. (Phys.-Math)
 MAKHORTOV S.D., Dr. Sci. (Phys.-Math)
 MANCIVODA A.V., Dr. Sci. (Phys.-Math)
 NAZIROV R.R. , Dr. Sci. (Tech)
 NECHAEV V.V., Cand. Sci. (Tech)
 NOVIKOV B.A., Dr. Sci. (Phys.-Math)
 PAVLOV V.L., USA
 PAL'CHUNOV D.E., Dr. Sci. (Phys.-Math)
 PETRENKO A.K., Dr. Sci. (Phys.-Math)
 POZDNEEV B.M., Dr. Sci. (Tech)
 POZIN B.A., Dr. Sci. (Tech)
 SEREBRJAKOV V.A., Dr. Sci. (Phys.-Math)
 SOROKIN A.V., Cand. Sci. (Tech)
 TEREKHOV A.N., Dr. Sci. (Phys.-Math)
 FILIMONOV N.B., Dr. Sci. (Tech)
 SHAPCHENKO K.A., Cand. Sci. (Phys.-Math)
 SHUNDEEV A.S., Cand. Sci. (Phys.-Math)
 SHCHUR L.N., Dr. Sci. (Phys.-Math)
 YAZOV Yu. K., Dr. Sci. (Tech)

Editors: LYSENKO A.V., CHUGUNOVA A.V.**CONTENTS**

Sidnyaev N. I., Butenko Yu. I., Bolotova E. E. Logical Model of Information and System Reliability Requirements for Knowledge Bases of Intellectual Systems	195
Bulgakov D. Yu., Bulgakova Ya. V., Karatygin N. A. Up-to-date Open-Source Software for the Analysis and Processing of Electroencephalograms: Opportunities and the Choices	205
Maltsev A. V. Methods for Distributed Rendering of Virtual Scenes on GPU with Motion Blur Implementation	213
Orlova E. V. Methods and Models of Data Analysis and Machine Learning in the Problem of Labor Productivity Management	219
Devyanin P. N., Leonova M. A. Application of Subtypes and Total Functions of Event-B Formal Method for the Formalization and Verification of the MROSL DP-Model	230
Cherepnev M. A. Estimation of the Speed of a New Parallel Block Algorithm for Finding Solutions of the Large Sparse Linear Systems over a Large Prime Field	242
Noskov S. I., Vergasov A. S Regression Model of Structural Factors of Cyber Threats	251

Н. И. Сидняев, д-р техн. наук, проф., зав. кафедрой, sidnyaev@bmstu.ru,
Ю. И. Бутенко, канд. техн. наук, доц., iuliiaabutenko2015@yandex.ru,
Е. Е. Болотова, магистр, bolotovaee@mail.ru,
Московский государственный технический университет им. Н. Э. Баумана

Логическая модель требований информационно-системной надежности для баз знаний интеллектуальных систем

Предложены логические модели представления знаний, сформированы требования информационно-системной надежности баз знаний интеллектуальных систем, обоснован комплекс показателей информационно-системной надежности базы знаний, проведен анализ особенностей ситуационного подхода при управлении и возможности его применения для мониторинга состояния базы знаний. Представлен облик системы мониторинга для обеспечения информационно-системной надежности интеллектуальной системы. Выделены и разработаны основные требования по обеспечению надежного поведения и работоспособности информационных систем.

Ключевые слова: база знаний, информационные системы, управление, надежность, мониторинг, информация, данные, знания

Введение

Системы искусственного интеллекта применяют во многих областях науки, так как с их помощью удается решать сложные задачи творческого характера. Интеллектуальные системы позволяют моделировать разумное поведение человека, имитировать процессы решения человеком сложных научно-технических задач. Применение систем искусственного интеллекта для поддержки принятия решений в сложных ситуациях востребовано в различных областях промышленности, особенно в тех, где нужно опираться на логически обоснованные, доказуемые аргументы. При принятии решений необходимы проведение детального моделирования с анализом последствий предполагаемого решения, поиск оптимального пути достижения заданного результата с помощью технологий имитационного моделирования. Как показано в работе [1], с подобными задачами успешно справляются экспертные системы — комплексы программных средств, способные частично или полностью заменять специалиста-эксперта при решении сложных задач, возникающих в процессе проектирования, диагностики и эксплуатации каких-либо технических систем.

В сфере проектирования экспертных систем проведено множество исследований, сосредоточенных на попытках совершенствования алгоритмов принятия решений, например, в работах [2, 3] описаны алгоритмы построения и оптимизации функциональной схемы системы поддержки принятия решений. Чтобы экспертные системы могли использовать большое количество знаний для решения сложных задач, возникла необходимость в создании баз знаний для описания различных предметных областей.

Ядром любой интеллектуальной системы является база знаний, которая описывает определенную предметную область. База знаний выступает в качестве хранилища фактов и правил, т. е. всех знаний о проблемной области, которые получают от разных специалистов определенной области знаний. Решение задачи представления знаний о предметной области предполагает их адекватное отображение в формализованном виде. Автоматизированная обработка таких знаний позволит эффективно решать задачи конкретной предметной области. Для этого используют специальные языки представления знаний, основанные на символьном представлении данных: фреймы, семантические сети, продукционные модели, логика предикатов.

Как показано в работах [4–6], выбор адекватной реалиям модели знания в системах искусственного интеллекта играет важное значение и в большинстве случаев становится основой выбора подхода к решению той или иной задачи. Помимо выбора необходимой модели представления знаний, важную роль играет поддержание логической и физической целостности базы знаний в процессе ее проектирования и эксплуатации, т. е. обеспечение надежной работы системы в условиях возможных сбоев технических и программных средств, ошибок пользователей.

В связи с отмеченными причинами актуальной и практически значимой задачей является построение информационной структуры для представления данных о предметной области, которая обеспечит эффективную работу экспертной системы в решении сложных задач, возникающих при проектировании, эксплуатации и диагностике каких-либо технических систем.

Логические модели представления знаний

Среди множества различных моделей представления знаний (рис. 1) в области обработки знаний наиболее востребована логика предикатов как одно из направлений эффективного вывода информации [7, 8].

Основная идея при построении логических моделей знаний заключается в том, что вся информация, необходимая для решения прикладных задач, рассматривается как совокупность фактов и утверждений, которые представляются как формулы в некоторой логике. Знания отображаются совокупностью таких формул, а получение новых знаний сводится к реализации процедур логического вывода. В качестве "фундамента" в логических моделях используется классический аппарат математической логики, методы которой достаточно хорошо изучены и формально обоснованы [7, 8]. Язык математической логики позволяет формально и в то же время доступно описывать понятия предметной области и связи между ними. Однако независимо от того, используется или не используется логика предикатов для представления знаний, она составляет теоретический фундамент в систематике описания, а по своему содержанию находится в основе всех пред-



Рис. 1. Модели представления знаний

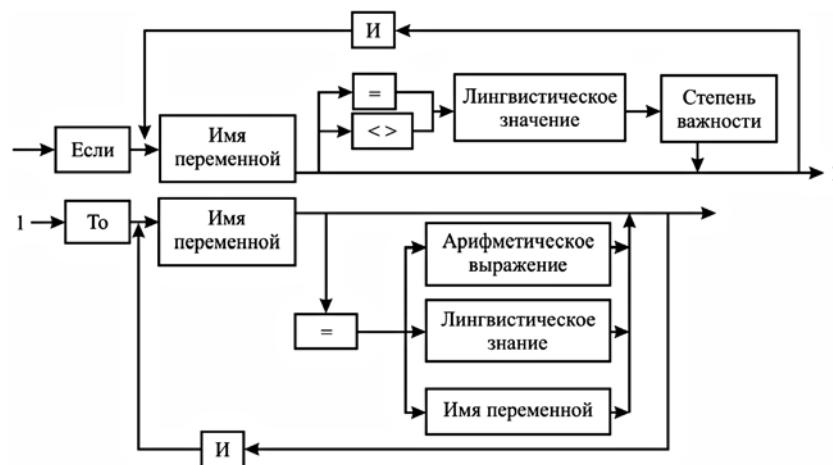


Рис. 2. Структура продукционного правила

ствлений знаний. Например, формализм *если — то* продукционных правил по виду (но не по семантике) аналогичен отношению импликации в логике предикатов (рис. 2); что же касается дуг (отношений), которые связывают узлы в семантической сети, то, если узлу поставить в соответствие терм, а дуге — предикатный символ, получаем предикат. Например, для представления вида "если A , то B " выполняется антитезис "если не B , то не A " (если $\sim B$, то $\sim A$). Следовательно, исходя из знаний вида " $\sim B$ " и "если A , то B ", на вопрос " $\sim A$?" должен последовать ответ "Да". Когда проводится формальный вывод методами символьной обработки, то такой результат получить нельзя, если положить, что A и $\sim A$ — различные объекты, однако до какой степени это соответствие учитывается в современных системах инженерии знаний? Если ответ на вопрос дается методом, при котором в выводе предполагается направленность, как это имело место в восходящих или нисходящих выводах, то для одного правила должны быть заданы все его эквивалентные выражения. Например, когда имеется правило "если A_1, A_2, \dots, A_n , то B ", то в качестве эквивалентных ему выражений должны выполняться правила "если $\sim B, A_2, \dots, A_n$ то $\sim A_1$ ", "если $A_1, \sim A_2, \dots, A_n$ то $\sim A_2$ ", ..., "если $A_1, A_2, \dots, \sim B$, то $\sim A$ " или же эти эквивалентные выражения должны

создаваться внутри механизма выводов. После создания исходная формула будет преобразована к следующему виду: $\sim A_1 \vee \sim A_2 \vee \dots \vee \sim A_n \vee B$. Это не что иное, как использование резолюции при доказательстве. Все это вызвано тем, что полнота метода резолюции нивелирует в импликационном отношении внешнее различие, базирующееся на положении антецедента и консеквента внутри логического выражения, а сам метод может использовать только ту информацию, которая семантически содержится внутри этого выражения. Так, используемый в качестве примера предикат "модели представления знаний (x, y)" описывает " x и y , которые находятся в отношении моделей представления знаний". Можно использовать систему с логикой предикатов второго порядка. Рассмотрим в соответствии с приведенным выше примером предикат "отношение (x, y, z)".

Положим, этот предикат говорит о том, что "отношение между u и z есть x ", и пусть предикат "модели представления знаний (y, z)", являющийся предикатом первого порядка, входит в него в виде предиката "отношение (модели представления знаний, y, z)". Если в этом случае записать $(\exists x)$ "отношение (x , фреймовая модель, семантическая сеть)", то с помощью данной предикатной формулы можно описать вопросительное предложение на естественном языке: "В каких отношениях находятся "фреймовая модель" и "семан-

тическая сеть?" Из предикатной формулы "отношение (модели представления знаний, y, z)" следует, что "фреймовая модель" и "семантическая сеть" находятся в отношениях "модели представления знаний". Существует множество самых различных отношений, которые могут возникать между y и z . Например, при подстановке отношений в x получаем отношение (модели представления знаний, y, z), отношение (системы искусственного интеллекта, y, z). Затем это перепишем в виде "отношение — классификация моделей представления знаний (y, z)", "отношение — классификация систем искусственного интеллекта (y, z)". Внутри выражения, относящегося к отношению (x, y, z), можно получить разложение выражения по переменной x в следующем виде:

$(\forall x)$ отношение (x, y, z) = отношение — классификация моделей представления знаний;

$(y, z) \wedge$ отношение — классификация систем искусственного интеллекта ($y, z) \wedge \dots$

$(\exists x)$ отношение (x, y, z) = отношение — классификация моделей представления знаний;

$(y, z) \vee$ отношение — классификация систем искусственного интеллекта ($y, z) \vee \dots$

Такое преобразование позволит дать описательные возможности предложений, которые будут достаточно велики и в рамках логики предикатов второго порядка.

Формирование требований информационно-системной надежности баз знаний интеллектуальных систем

Вопрос надежности является главным в системном подходе к проектированию баз знаний интеллектуальных систем. Для обеспечения нормального функционирования системы искусственного интеллекта и высокой эффективности обработки данных необходимо поддержание ее системной и информационной надежности, что позволит снизить ошибочность, неоднозначность, неполноту и противоречивость исходных данных в базе знаний. Надежность — свойство, характеризующее способность системы удерживать в условиях возмущений свое состояние, близкое к невозмущенному [9, 10]. Информационная надежность обеспечивает целостность и непротиворечивость данных в базе знаний, а системная надежность обеспечивает надежность системы как сложного технического или производственного объекта [11].

Представление объекта исследования в виде системы базы знаний позволяет воспользоваться методами системного анализа в целях изучения его свойств. Среди наиболее важных системных свойств выделим свойства надежности, управляемости и наблюдаемости, которые достаточно хорошо исследованы в применении, в первую очередь, к объектам, моделируемым в виде динамических систем [12].

Рассмотрим динамическую систему с переходной функцией состояния:

$$g(t) = \varphi[t; g(t_0), u_{[t_0, t]}],$$

где $g(t)$ — вектор-функция состояния системы (матрица-столбец размерности $r \times 1$); r — число строк; φ — переходная вектор-функция состояний; $g(t_0)$ — состояние системы в начальный момент времени t_0 ; $u(t)$ — вектор-функция входных воздействий, определенная на отрезке времени $[t_0, t]$.

Пусть известна параметрически заданная траектория $g(t)$ в пространстве состояний при конкретных $\tilde{g}(t)$. Динамическая система будет устойчивой [12], если при любом значении $\varepsilon > 0$ существует отличное от нуля расстояние $\delta > 0$ между их начальными условиями $\rho_0[\tilde{g}(t_0), g(t_0)] \leq \delta$ такое, что расстояние $\rho_t[\tilde{g}(t), g(t)]$ между возмущенной $\tilde{g}(t)$ и невозмущенной $g(t)$ траекториями в момент времени $t > t_0$ будет $\rho_t < \varepsilon$. Другими словами, если в некоторый момент времени существует такое отклонение реальной траектории процесса от идеальной, которое гарантирует, что отклонение реальной траектории от идеальной во все время после начального будет в заданных пределах, то рассматриваемая система надежна. Здесь под расстоянием между траекториями можно понимать следующую метрику:

$$\rho_t[\tilde{g}(t), g(t)] = \left[\sum_i^r (\tilde{g}_{t_i} - g_{t_i})^2 \right]^{1/2},$$

где r — число координат, или параметров состояния.

Если на момент времени $t > t_0$ траектория возмущенного движения укладывается в окрестность заданного радиуса ε , то это обстоятельство означает, что динамическая система устойчива. Окончательное заключение можно сделать тогда, когда для любого малого значения ε существует такое начальное возмущение траектории, при котором всякая возмущенная траектория движения с допустимым возмущением в начальный момент будет и в дальнейшем отличной от невозмущенной с заданным отклонением.

Еще более сложный механизм обеспечения надежного поведения наблюдается у целенаправленных систем, ориентированных на выполнение строго определенных целей, которые формируются внутри системы [13, 14]. Комплексный показатель надежности характеризует несколько свойств, определяющих надежность системы, и обеспечивает надежность функционирования информационных систем, которые рассматриваются как сложные человеко-машические системы, включающие в свой состав эргатические звенья, технические средства и программное обеспечение. Все методы обеспечения надежности и достоверности информационных систем можно отнести к двум классам. Один класс включает в себя методы, обеспечивающие безошибочность (безотказность) функциональных технических, эргатических и программных звеньев, т. е. повышает их надежность. Другой класс — методы, обеспечивающие обнаружение и исправление ошибок, возникающих в информации, т. е. методы контроля достоверности информации и ее коррекции, косвенно также повышающие функциональную надежность систем.

Существуют различные виды обеспечения надежности: экономическое, временное, организационное,

структурное, технологическое, эксплуатационное, социальное, эргатическое, алгоритмическое, синтаксическое, семантическое. Обеспечение можно определить как совокупность факторов (элементов, методов, приемов, процедур, ресурсов и т. п.), способствующих достижению поставленной цели. Организационное, эксплуатационное, техническое, социальное и эргатическое обеспечения применяют преимущественно для повышения надежности систем, а структурное и алгоритмическое обеспечение — для обоих классов методов.

В связи с этим важным свойством системы является управляемость, характеризующее принципиальную возможность перехода ее из начального состояния в заданное [13, 14]. Для реализации свойства управляемости требуется однозначное соответствие между входными параметрами и параметрами системы. Набор управляющих воздействий (причинно-следственных связей) между входными параметрами и параметрами системы) должен позволять достигать цели управления. В общем случае система как объект управления должна однозначно реагировать на управляющие воздействия и эта реакция должна однозначно наблюдаться. По этой причине следующим важным свойством является наблюдаемость системы.

Наблюдаемость — свойство системы, характеризующее принципиальную возможность определения состояния системы по ее выходным параметрам [11]. Для реализации свойства требуется однозначное соответствие между параметрами системы и выходными параметрами.

Следует отметить, что главной особенностью всех известных методов анализа свойств систем является допущение, что модель системы известна достаточно полно, т. е. решена задача идентификации системы.

Понятие идентифицируемости относится к вопросу о единственности представления выбранного системного описания в рамках структуры модели.

Под идентифицируемостью системы понимается возможность построения процедуры идентификации, обеспечивающей единственность представления данного системного описания в рамках выбранной структуры модели, т. е. обеспечение обратимости отображения вход — выход. Система будет идентифицируемой, если процедура идентификации приводит к однозначному определению структуры модели и ее параметров. Под идентификацией системы понимается процесс построения или выбора классической математической модели системы по результатам наблюдения (измерения) входных и выходных параметров объекта [11, 13]. В общем случае необходимыми условиями идентифицируемости являются управляемость и наблюдаемость [15].

Таким образом, системные свойства управляемости, наблюдаемости и идентифицируемости являются связанными.

Рассмотрим взаимосвязь этих свойств со свойством надежности системы. Пусть система S первоначально представляет собой "черный ящик" или систему типа "вход — выход", которую можно формально задать как подмножество декартова произведения множества входных параметров X и множе-

ства выходных параметров Y (т. е. как подмножество множества всех упорядоченных пар (x, y) таких, что $x \in X, y \in Y: S \subset X \times Y$). В случае, когда S является функцией $S: X \rightarrow Y$, система будет функциональной системой. Здесь $X = \{x_i\}, i = 1, n$, где n — множество входных параметров, которые определяют влияние среды на систему, или выходных параметров: $Y = \{y_j\}, j = 1, m$, где m — множество выходных параметров, через которые сама система влияет на среду. Множество всех возможных пар $S = \{x_i, y_j\}$, образующих декартово произведение и представляющее собой множество причинно-следственных связей, является бинарными отношениями $s_{ij}: x_i \rightarrow y_j$. Тогда $S = \{s_{ij}\}$, причем бинарное отношение на множествах X и Y может быть задано графически.

Если пара (x_i, y_j) принадлежит отношению S , соединяя изображенные точки x_i и y_j линией, направленной от первого элемента пары ко второму. Направленные линии, соединяющие пары точек, есть дуги, а точки обозначают элементы множеств — вершины графа. При этом множество всех функций $S = \{s_{ij}\}$ может быть декомпозировано на два подмножества: множество известных, штатных функций $S_{ш}$, описывающих явные причинно-следственные связи, известные для наблюдателя, и множество неизвестных, нештатных функций $S_{нш}$, описывающих неявные причинно-следственные связи, неизвестные наблюдателю. Так, если функция $s_{32} \in S_{ш}$, а функция s_{21} известна не полностью, тогда $s_{21} \in S_{нш}$.

Дальнейшим развитием вариантов представления системы является так называемый "серый ящик" [16]. Это вариант описания систем с учетом внутреннего состояния системы. В этом случае вводится дополнительный объект $G = \{g_k, k = 1, r\}$, где G — множество внутренних состояний системы; g_k — параметры состояний, и система S уже представляется совокупностью преобразований.

В частном случае, когда S задается двумя функциями $S: \{X \rightarrow G; X \times G \rightarrow Y\}$, систему называют автоматом [12]. Для динамической системы в описание включается множество моментов времени $T = \{t\}$ и параметры системы являются функциями времени: $x(t), g(t), y(t)$. Таким образом, задана система:

$$S \subset X \times G \times Y \text{ или } S: \{X \rightarrow G; X \times G \rightarrow Y\},$$

$$\Delta Y(t) = Y(t) - Y_{\text{уст}}$$

Здесь $\Delta Y(t)$ — рассогласование для $Y_{\text{уст}}$ — множества установленных значений выходных параметров. Реализация обратной связи предполагает оценку рассогласования выходных параметров и выработку дополнительных управляющих воздействий на вход системы $\Delta X(t + 1) = F[\Delta Y(t)]$. Тогда с учетом сигнала обратной связи:

$$X_{\text{oc}}(t + 1) = X(t + 1) + \Delta X(t + 1),$$

$$Y(t + 1) = S\{X_{\text{oc}}(t + 1); G\} = S\{X(t + 1) + \Delta X(t + 1); G\} = S\{X(t + 1) + F[\Delta Y(t)]; G\}.$$

Успешная реализация представленного соотношения предполагает знание операторов S и F . При этом оператор S должен обеспечить возможность управле-

ния системой, т. е. отображения $X \rightarrow G$ и наблюдения ее состояния $G \rightarrow Y$, т. е. выполнения требований управляемости и наблюдаемости.

Следует отметить, что условием надежности системы с обратной связью является ее идентифицируемость, т. е. знание модели системы. Действительно, если система не будет идентифицируема, то она или неуправляема, или не наблюдаема, или же неуправляема и ненаблюдаема одновременно. Это означает, что отображения $X \rightarrow G$ и $G \rightarrow Y$ могут быть реализованы некорректно и результат $\Delta Y(t)$ будет соответствовать текущему поведению системы. Таким образом, знание оператора S предполагает идентифицируемость системы, т. е. знание модели ее поведения. Требования к оператору F вытекают из требований однозначности выработки дополнительного управляющего воздействия $\Delta X(t+1)$ на основании известного рассогласования $\Delta Y(t)$. При этом следует также:

- оценить текущее состояние системы G^* ;
- выработать варианты реакций $\{\Delta X\}$ на рассогласование ΔY ;
- осуществить прогноз последствий принимаемых решений;
- привести выбранное воздействие $\Delta Y(t)$ к требуемому виду.

С учетом многообразия решаемых задач такая информационная система может быть отнесена к классу систем мониторинга с расширенным перечнем функций, касающихся вопросов прогнозирования и управления с предсказанием ввиду инерционности объекта управления. Важной составляющей процесса мониторинга является оценивание текущего состояния системы G^* , т. е. решение задачи идентификации системы. В соответствии с выдвинутым ранее утверждением надежность системы невозможна без знания модели системы.

Обоснование комплекса показателей информационно-системной надежности базы знаний

Показатель надежности представляет собой техническую характеристику, определяющую одно или несколько свойств, составляющих надежность объекта [10]. Охарактеризуем комплекс показателей информационно-системной надежности баз знаний. Будем исходить из следующего перечня требований к заданию различного рода показателей. Показатель должен быть:

1) физически измеримой предметной величиной, физическая измеримость должна сочетаться с вычислимостью на основании эмпирических данных;

2) скалярной величиной, универсальным, т. е. применимым к системе в целом, к подсистемам и компонентам;

3) стабильным, постоянным на всех этапах жизненного цикла системы — от замысла до замены, а также оперативным, не требовать длительного времени для получения оценки по нему;

4) прогностичным, интуитивно приемлемым к ретроспективе и текущему моменту, перспективе,

а также не создавать трудностей в восприятии и понимании;

5) пригодным и положительным для управления системой, т. е. ситуационным, и отображать тенденцию к развитию системы.

Рассмотрим ситуацию, когда задается общее число решаемых задач $N_3^{\text{общ}}$ при известном числе конкретных ситуационных задач типа:

- выполненных с требуемым качеством ($N_3^{\text{вып}}$);
- не выполненных с требуемым качеством ($N_3^{\text{н/вып}}$);
- информационно не обеспеченных и некорректно решенных ($N_3^{\text{некорр}}$);
- при выполнении которых возникли конфликты ($N_3^{\text{конф}}$).

Отсутствие управления базами знаний может привести к существенным временным и финансовым потерям. Наличие управляемой базы знаний позволяет сохранить полученный опыт и наработки важных показателей. Здесь необходимо отметить предназначение каждого показателя в процессах управления базами знаний [17].

Так, например, целевой процесс означает, что каждая задача жизненного цикла системы должна быть выполнена. Следовательно, если задано общее число задач $N_3^{\text{общ}}$ и число задач, выполненных с требуемым качеством $N_3^{\text{вып}}$, то можно ввести относительный показатель выполнения задач, т. е. показатель выполнения всего целевого процесса:

$$\eta_{\text{цел}} = \frac{N_3^{\text{вып}}}{N_3^{\text{некорр.общ}}} N_3^{\text{общ}}.$$

Тогда критерий базы знаний по целевому процессу должен удовлетворять требованию $\eta_{\text{цел}} \geq \eta_{\text{цел}}^{\text{треб}}$.

Сенсорный процесс означает, что каждая из задач должна быть обеспечена соответствующей информацией I_{3j} , причем общий объем информации представляется в виде $I_{\text{общ}} = \cup \{I_{3j}\}$. Исходя из этого возможный вариант показателя будет определяться общим числом произведения задач $N_3^{\text{некорр.общ}} N_3^{\text{общ}}$ и числом информационно необеспеченных и некорректно решенных задач $N_3^{\text{некорр}}$. Тогда показатель качества сенсорного процесса представляется как показатель информационной обеспеченности:

$$\eta_{\text{сенс}} = 1 - \frac{N_3^{\text{некорр}}}{N_3^{\text{общ}}}.$$

Возможен частный вариант в случае использования вместо $N_3^{\text{общ}}$ числа нерешенных (невыполненных) задач $N_3^{\text{н/вып}}$. Тогда

$$\eta_{\text{сенс}} = 1 - \frac{N_3^{\text{некорр}}}{N_3^{\text{н/вып}}}.$$

Критерий базы знаний по сенсорному процессу, по отношению к требуемому $\eta_{\text{сенс}}^{\text{треб}}$, выглядит следующим образом:

$$\eta_{\text{сенс}} \geq \eta_{\text{цел}} \geq \eta_{\text{сенс}}^{\text{треб}}.$$

Процесс сертификации (гомеостаза) — процесс устранения конфликтов, причинами которых могут быть следующие факторы: несоответствие входных и выходных данных, временных параметров; невыполнение требований по качеству обработки или решения задач; нехватка ресурсов и т. д. [13]. Если известно общее число задач ($N_3^{\text{общ}}$) и число задач, при выполнении которых возникли конфликты ($N_3^{\text{конф}}$), то показатель сертификации как показатель бесконфликтности будет выглядеть следующим образом:

$$\eta_{\text{сенс}} = 1 - \frac{N_3^{\text{конф}}}{N_3^{\text{общ}}}.$$

Тогда критерий баз знаний по процессу сертификации $\eta_{\text{цел}} \geq \eta_{\text{треб}}$. Следует учитывать, что все показатели взаимосвязаны между собой:

$$\eta_{\text{сенс}} = 1 - \eta_{\text{цел}} \frac{N_3^{\text{некорр}}}{N_3^{\text{вып}}},$$

$$\eta_{\text{серт}} = 1 - \eta_{\text{цел}} \frac{N_3^{\text{конф}}}{N_3^{\text{вып}}}.$$

Приведенные выше показатели позволяют проиллюстрировать графически выполнение требований баз знаний для реальной системы и могут быть использованы в экспертных системах при принятии решений. Так, например, когда максимальные значения показателей не превышают единицу и заданы требования по каждому из них, можно указать область надежности системы, в которой выполняются требования для баз знаний.

Рассмотрим взаимосвязь между показателями несколько подробнее. Пусть выполняется условие $N_3^{\text{вып}} = N_3^{\text{общ}} - (N_3^{\text{некорр}} + N_3^{\text{конф}})$, тогда

$$\eta_{\text{цел}} = \frac{N_3^{\text{общ}} - (N_3^{\text{некорр}} + N_3^{\text{конф}})}{N_3^{\text{общ}}},$$

или с учетом показателей сенсорного и сертификационного процессов

$$\eta_{\text{цел}} = \frac{N_3^{\text{общ}} - N_3^{\text{общ}}(1 - \eta_{\text{сенс}}) - N_3^{\text{общ}}(1 - \eta_{\text{серт}})}{N_3^{\text{общ}}},$$

$$\eta_{\text{цел}} = \eta_{\text{сенс}} + \eta_{\text{серт}} - 1.$$

Полученное выражение позволяет оценить требования к процессам для сохранения требований баз знаний. Для равных значений $\eta_{\text{сенс}}$ и $\eta_{\text{серт}}$, изменяющихся в пределах от нуля до единицы, показана область приемлемых значений $\eta_{\text{цел}}$ для высоких уровней значений. Например, в практике эксплуатации промышленных систем уровень надежности задается как вероятность безотказной работы $P_{\text{бр}} \geq 0,9$, а в ряде случаев для критически важных систем — как $P_{\text{бр}} \geq 0,999$. Если заданы требования по показателю $\eta_{\text{цел}}^{\text{треб}}$, то можно определить и требования по другим показателям как

$$1 - \eta_{\text{цел}}^{\text{треб}} = \eta_{\text{сенс}} + \eta_{\text{серт}},$$

$$1 - \eta_{\text{цел}}^{\text{треб}} = \eta_{\text{сенс}} + \eta_{\text{серт}}.$$

Как видно, при высоких значениях $\eta_{\text{цел}}^{\text{треб}}$ крайне важное значение приобретают вопросы информационной обеспеченности и бесконфликтности системы: $(\eta_{\text{сенс}} + \eta_{\text{серт}}) < 0,1$. Указанные вопросы тесно связаны с вопросами корректности решения задач и проектирования систем. В области информационных систем, неизбежно использующих программные комплексы, указанные вопросы также связаны с разделом



Рис. 3. Иерархическая структура показателей

программной инженерии. Введенные выше показатели баз знаний позволяют представить комплекс показателей промышленных систем в виде иерархической структуры. Представляется, что на верхнем уровне иерархии должны быть общесистемные показатели, инвариантные к природе системы; на следующем уровне — показатели, характеризующие проблемную область и назначение системы; далее — локальные показатели, отражающие свойства отдельных составляющих предметной области. Пример структуры приведен на рис. 3.

Анализ особенностей ситуационного подхода к управлению и возможности его применения для мониторинга состояния базы знаний

В современном мире информационных технологий базу знаний описывают как совокупность программных средств, которые обеспечивают хранение, поиск, запись и преобразование структурированных информационных единиц. Знания (информационные единицы) рассматривают как совокупность сведений, которые образуют целостное описание, соответствующее определенному уровню осведомленности об указанном предмете, проблеме, событии [18]. Важным в настоящее время является обеспечение баз знаний поведенческой системой, которая предполагает наличие контура обратной связи, составной частью которого является система мониторинга состояния поведенческой системы. Создание такой системы мониторинга предполагает выбор подхода, позволяющего учесть особенности самого объекта мониторинга как поведенческой системы, ее цели и задачи, а также особенности условий ее функционирования. Наиболее приемлемым в данном случае является ситуационный подход, который предоставляет необходимое условие рефлексивного интеллектуального управления, когда рассматриваются вопросы создания интеллектуальных автоматизированных обучающих систем с функцией рефлексии [19, 20]. При использовании теории рефлексии в структуру планировщика обучающей системы вводится условно интеллектуально рефлексивный агент, обеспечивающий поиск компромиссных решений при взаимодействии моделей ученика и учителя.

Ключевым этапом интеллектуального принятия решений является анализ особенностей ситуации, а сам процесс управления может быть представлен как непрерывный процесс реагирования системы на возникающие ситуации. При этом не существует единого решения для всех ситуаций, а решение, оптимальное для одной ситуации, может быть полностью непригодным для другой. Ситуационный подход к интеллектуальному управлению предполагает осуществление непрерывного наблюдения за объектом управления, который рассматривается неотрывно от окружающей его и взаимодействующей с ним проблемной области [20]. При этом в любой момент времени анализируемый объект описывается

двумя типами состояний, формирующих текущую ситуацию: текущим, т. е. реальным в данный момент, и целевым — требуемым с точки зрения субъекта управления, лица, принимающего решения (ЛПР). Несоответствие текущего и целевого состояний рождает необходимость принятия и реализации соответствующих решений по достижению последнего. Такие ситуации называют проблемными. В зависимости от особенностей процесса управления могут быть различные критерии и показатели отнесения ситуации к числу проблемных.

Характерной чертой данной концепции является априорное отсутствие формализованного закона управления, позволяющего выбрать наилучшее для достижения целевого состояния решение, что, в свою очередь, вызвано возможностью изменения цели управления. Формализация закона управления и постановка задачи на выбор управляющего воздействия осуществляются на основании анализа особенностей рассматриваемой ситуации. При этом в рамках ситуационного управления широкое распространение получили методы анализа ситуаций с заблаговременным разделением ситуаций на классы. Указанные методы предполагают, что на основании имеющейся информации об особенностях объекта управления, а также существующих или накопленных в процессе управления опытных данных могут быть заблаговременно сформированы готовые решения или способы, схемы их получения для определенных групп ситуаций, образующих отдельный класс.

При этом если в первом случае предполагается использование различных шаблонов, позволяющих описать группы ситуаций, то во втором случае предполагается объединение предполагаемых ситуаций, схожих между собой по различным признакам, в непересекающиеся классы. С точки зрения ситуационного подхода выделяют основные этапы принятия управленческих решений и проводят поиск известных прецедентов разрешения проблемных ситуаций. Порядок выявления проблемных ситуаций зависит от используемых целевых показателей и критерии, в качестве которых могут выступать показатели эффективности и качества, при этом наиболее часто используемым критерием является критерий пригодности вида

$$K^{\text{пр}} = \begin{cases} Y_{\text{пр}}, \Pi < \Pi_{\text{тр.}} \\ Y_{\text{пр}}, \Pi \geq \Pi_{\text{тр.}} \end{cases},$$
$$K^{\text{пр}} = \{Y_{\text{пр}}\},$$

где $Y_{\text{пр}}$ — суждение о том, что рассматриваемая ситуация является проблемной, при выполнении условий; Π , $\Pi_{\text{тр.}}$ — соответственно текущее и требуемое значения некоторого целевого показателя. Если ситуация является проблемной, то в целях ее разрешения осуществляется поиск прецедентов, схожих с ситуацией, которые известны ЛПР [19, 20].

Диагностика проблемных ситуаций осуществляется в рамках реализации одного из основных принципов ситуационного подхода — непрерывно-

го наблюдения за состоянием объекта управления. Целью диагностики является заблаговременное обнаружение негативных тенденций в функционировании системы, а также внешней среды, способных привести к проблемным ситуациям.

Облик системы мониторинга сложной системы для обеспечения информационно-системной надежности

Для обеспечения баз знаний поведенческой системы находящаяся в контуре обратной связи система мониторинга должна быть дополнена комплексом моделей предметной области. Выделим следующие модели, которые должны присутствовать в системе мониторинга.

1. Общая модель системы в виде оператора преобразования множества задач системы Φ во множество результатов их выполнения представляется в виде $\Lambda = \{\lambda\}S_1 : \phi \xrightarrow{\Sigma} \Lambda$.

Здесь в основе лежит достаточно известный аппарат математической логики. Процесс доказательства основывается на логике высказываний и логике предикатов. С точки зрения выразительности ее можно охарактеризовать как классическую логику нулевого порядка [7].

2. Модель сбора информации об анализируемой системе, ее обработке и анализе состояния представляется в виде оператора преобразования множества исходных локальных показателей $\Pi_{\text{мл}} \xrightarrow{L_{\text{обр}}} M$ во множество состояний системы $M : S_2 : \Pi_{\text{мл}} \xrightarrow{L_{\text{обр}}} M$, где $L_{\text{обр}} = \langle L_{\text{изм}}, L_{\text{сб}}, L_{\text{обр}}, L_{\text{ан}} \rangle$ — оператор технологии обработки показателей, состоящий из компонент-операторов проведения измерений $L_{\text{изм}}$, сбора $L_{\text{сб}}$, обработки $L_{\text{обр}}$ и анализа $L_{\text{ан}}$.

3. Модель системы прогнозирования состояния и выработки стратегий управления состоянием поведенческой системы в виде оператора преобразования множества состояний во множество стратегий управления $\{U\} : S_2 : M \xrightarrow{L_{\text{упр}}} U$, где $L_{\text{упр}} = \langle L_{\text{прг}}, L_{\text{стр}} \rangle$ — технология управления состоянием поведенческой системы как совокупность операторов прогнозирования поведенческого состояния $L_{\text{стр}}$.

4. Модель системы оценки рисков выбранных стратегий и реализации приемлемых стратегий в виде оператора преобразования множества стратегий управления состоянием поведенческой системы $\{U\}$ во множество приемлемых на основе оценки рисков стратегий U^* ; $S_3 : U \xrightarrow{L_{\text{возд}}} U^*$, где $L_{\text{возд}} = \langle L_{\text{риск}}, L_{\text{выб}}, L_{\text{реал}} \rangle$ — технология воздействия на поведенческие системы базы знаний для приведения ее в заданное состояние как совокупность операторов оценки рисков возможных стратегий $L_{\text{риск}}$, выбора приемлемых стратегий $L_{\text{выб}}$, принятия решения и реализации выбранной стратегии $L_{\text{реал}}$.

Поведенческая система в целом для обеспечения надежности функционирования представляется как система с обратной связью, где используется система мониторинга, позволяющая реализовать требования базы знаний как требование надежности сложных информационных систем.

Учитывая значительную неопределенность происходящих процессов, многообразие вариантов поведения и объективную возможность возникновения нештатных ситуаций, система мониторинга является ситуационным центром, дополненным моделирующим комплексом.

Сituационный центр реализует следующие алгоритмы:

1) $L_{\text{обр}} = \langle L_{\text{изм}}, L_{\text{сб}}, L_{\text{обр}}, L_{\text{ан}} \rangle$ — алгоритм обработки показателей поведенческой системы как совокупность операторов производства измерений $L_{\text{изм}}$, сбора $L_{\text{сб}}$, обработки $L_{\text{обр}}$ и анализа $L_{\text{ан}}$;

2) $L_{\text{упр}} = \langle L_{\text{прг}}, L_{\text{стр}} \rangle$ — алгоритм управления состоянием поведенческой системы как совокупность операторов прогнозирования $L_{\text{прг}}$ и выработки стратегий управления состоянием поведенческой системы $L_{\text{стр}}$;

3) $L_{\text{возд}} = \langle L_{\text{изм}}, L_{\text{риск}}, L_{\text{выб}}, L_{\text{реал}} \rangle$ — алгоритм воздействия на поведенческие системы для приведения ее в заданное состояние как совокупность операторов оценки рисков возможных стратегий $L_{\text{риск}}$, выбора приемлемых стратегий $L_{\text{выб}}$, принятия решения и реализации выбранной стратегии $L_{\text{реал}}$.

Таким образом, общий алгоритм обеспечения функционирования баз знаний поведенческой системы будет выглядеть следующим образом:

$$L_{\text{ИСБ}} = \langle L_{\text{обр}}, L_{\text{упр}}, L_{\text{возд}} \rangle = \\ = \langle L_{\text{изм}}, L_{\text{сб}}, L_{\text{обр}}, L_{\text{ан}} \rangle \langle L_{\text{прг}}, L_{\text{стр}} \rangle \langle L_{\text{изм}}, L_{\text{риск}}, L_{\text{выб}}, L_{\text{реал}} \rangle.$$

Моделирующий комплекс в составе ситуационного центра предназначен для обоснования принимаемых решений и позволяет решить следующие задачи моделирования: жизненного цикла объекта исследований, жизненного цикла объектов предметной области, функционирования анализируемой поведенческой системы и ее предметной области. Сфера поведенческих интеллектуальных систем постоянно расширяется. Повсеместная информатизация приводит к накоплению огромных объемов данных в науке, производстве, бизнесе, транспорте, здравоохранении. Возникающие при этом задачи прогнозирования, управления и принятия решений часто сводятся к обучению по прецедентам. Можно заметить, что имеется множество объектов (ситуаций) и множество возможных ответов (откликов, реакций). Существует некоторая зависимость между ответами и объектами, но она не известна. Известна только конечная совокупность прецедентов — пар "объект, ответ", называемая обучающей выборкой. На основе этих данных требуется восстановить за-

висимость, т. е. построить алгоритм, способный для любого объекта выдать достаточно точный ответ. Для измерения точности ответов вводится операторный функционал. Данная постановка является обобщением классических задач аппроксимации функций. В классических задачах аппроксимации объектами являются действительные числа или векторы. В реальных прикладных задачах входные данные об объектах могут быть неполными, неточными, нечисловыми, разнородными. Эти особенности приводят к большому разнообразию ситуационных алгоритмов.

Заключение

Описаны разработанные требования к поддержанию логической и физической целостности баз знаний систем искусственного интеллекта. Предложена логика предикатов в качестве модели, обеспечивающей информационную надежность интеллектуальной системы, т. е. целостность и непротиворечивость данных в базе знаний. Выделены и описаны основные требования по обеспечению надежного поведения и работоспособности информационных систем: управляемость, наблюдаемость, идентифицируемость. Разработан и обоснован комплекс показателей информационно-системной надежности баз знаний. Проведен анализ особенностей ситуационного подхода к управлению базами знаний и возможности применения данного подхода для мониторинга состояния базы знаний. Показано, что наилучшая информационно-системная надежность достигается путем применения автоматизированных систем мониторинга для постоянного наблюдения и периодического анализа объектов системы с отслеживанием динамики происходящих изменений. Постулируется, что ситуационный подход в системе мониторинга позволяет заблаговременно обнаруживать негативные тенденции в функционировании системы.

Показано, что интеллектуальная система представляет собой набор информационных технологий, направленных на поддержку жизненного цикла информации и включающих три основных процесса: обработку данных, управление информацией и управление базами знаний. В условиях резкого увеличения объемов информации переход к работе со знаниями на основе искусственного интеллекта является, по всей вероятности, доминирующим фактором информационного общества.

Список литературы

- Чувиков Д. А. Применение экспертного моделирования в получении новых знаний человеком // Радиопромышленность. — 2017. — Т. 27, № 2. — С. 72—80.
- Курейчик В. М. Особенности построения систем поддержки принятия решений // Известия Южного федерального университета. Технические науки. — 2012. — Т. 132, № 7. — С. 92—98.
- Симанков В. С., Владимиров С. Н., Денисенко А. О. и др. Методологические аспекты построения систем поддержки принятия решений // Вестник Донского государственного технического университета. — 2018. — Т. 8, № 3. — С. 258—267.
- Сидняев Н. И., Бутенко Ю. И., Болотова Е. Е. Экспертная система продукционного типа для создания базы знаний о конструкциях летательных аппаратов // Аэрокосмическое приборостроение. — 2019. — № 6. — С. 38—52.
- Бояркина А. К., Ермолаева В. В. Экспертные системы // Молодой ученый. — 2016. — № 11. — С. 286—289.
- Елисеев Д. В. Модель представления знаний при создании адаптивной информационной системы // Наука и образование: научное издание МГТУ им. Н. Э. Баумана. — 2010. — № 3. — С. 1—6.
- Bruynooghe M., Blockeel H., Bogaerts B. et al. Predicate logic as a modeling language: modeling and solving some machine learning and data mining problems with IDP3 // Theory and Practice of Logic Programming. — 2015. — Vol. 15, No. 6. — P. 783—817.
- Страшова И. И. Модели представления знаний и особенности логической модели // Современные научные исследования и разработки. — 2017. — Т. 2, № 1. — С. 208—210.
- Лаврищева Е. М., Зеленов С. В., Пакулин Н. В. Методы оценки надежности программных и технических систем // Труды института системного программирования РАН. — 2019. — Т. 31, № 5. — С. 95—108.
- Сидняев Н. И. Статистический анализ и теория планирования эксперимента. — М.: Изд-во МГТУ им. Н. Э. Баумана, 2017. — 195 с.
- Потюпкин А. Ю., Чечкин А. В. Искусственный интеллект на базе информационно-системной избыточности. — М.: КУРС, 2019. — 384 с.
- Дружинина О. В., Масина О. Н. Методы анализа устойчивости динамических систем интеллектуального управления. — М.: URSS, 2016. — 248 с.
- Васенин В. А., Пирогов М. В., Чечкин А. В. Информационно-системная безопасность критических систем. — М.: КУРС, 2018. — 352 с.
- Васенин В. А., Пирогов М. В., Чечкин А. В. Основной оператор радикального моделирования на демонстрационном примере // Программная инженерия. — 2016. — Т. 7, № 2. — С. 75—85.
- Зыбин Е. Ю. Об идентифицируемости линейных динамических систем в замкнутом контуре в режиме нормальной эксплуатации // Известия ЮФУ. Технические науки. — 2015. — № 4 (165). — С. 160—170.
- Сидняев Н. И., Храпов П. В. Нейросети и нейроматематика / под ред. Н. И. Сидняева. — М.: Изд-во МГТУ им. Н. Э. Баумана, 2016. — 83 с.
- Дьяченко Р. А., Махаммад М. Д. К вопросу об оценке надежности систем управления базами данных // Вестник ИрГТУ. — 2009. — № 1 (37). — С. 196—199.
- Палюх Б. В., Бурдо Г. Б. Представление знаний в системах автоматизированного проектирования и управления технологическими процессами // Вестник ТГТУ. — 2010. — № 2. — С. 258—265.
- Фридман А. Я. Ситуационное моделирование иерархических динамических систем // Информационные и математические технологии в науке и управлении. — 2018. — № 1 (9). — С. 5—15.
- Филимонов А. Б., Филимонов Н. Б. Ситуационный подход в задачах автоматизации управления техническими объектами // Мехатроника, автоматизация, управление. — 2018. — Т. 19, № 9. — С. 563—578.

Logical Model of Information and System Reliability Requirements for Knowledge Bases of Intellectual Systems

N. I. Sidnyaev, sidnyaev@bmstu.ru, **Yu. I. Butenko**, iuliiabutenko2015@yandex.ru,
E. E. Bolotova, lisayshka_bolotova@mail.ru, Moscow State Technical University, Moscow,
105005, Russian Federation

Corresponding author:

Bolotova Elizaveta E., Master, Moscow State Technical University, Moscow, 105005, Russian Federation
E-mail: lisayshka_bolotova@mail.ru

Received on April 16, 2020

Accepted on May 20, 2020

The paper proposes a logical model of knowledge representation, forms requirements for information and system reliability of knowledge bases of intelligent systems, justifies a set of indicators of information-system reliability knowledge base, provides the analysis of the characteristics of the situational approach in management and its application for condition monitoring knowledge base. The article presents the image of the monitoring system to ensure information and system reliability of the intellectual system. The main requirements for ensuring reliable behavior and operability of information systems are identified and developed.

Keywords: knowledge base, information systems, control, reliability, monitoring, information, data, knowledge

For citation:

Sidnyaev N. I., Butenko Yu. I., Bolotova E. E. Logical Model of Information and System Reliability Requirements for Knowledge Bases of Intellectual Systems, *Programmnaya Ingeneria*, 2020, vol. 11, no. 4, pp. 195–204

DOI: 10.17587/prin.11.195-204

References

1. Chuvikov D. A. Application of expert modeling in obtaining new knowledge by a person, *Radiopromyshlennost'*, 2017, vol. 27, no. 2, pp. 72–80 (in Russian).
2. Kureychik V. M. Features of construction of systems of support of decision-making, *Izvestiya Yuzhnogo federal'nogo universiteta. Tekhnicheskie nauki*, 2012, vol. 132, no. 7, pp. 92–98 (in Russian).
3. Simankov V. S., Vladimirov S. N., Denisenko A. O. et al. Methodological aspects of building decision support systems, *Vestnik Donskogo gosudarstvennogo tekhnicheskogo universiteta*, 2018, vol. 8, no. 3, pp. 258–267 (in Russian).
4. Sidnyaev N. I., Butenko Yu. I., Bolotova E. E. Expert system of a production type for creation of a knowledge base about the aircraft constructions, *Aerospace instrument making*, 2019, no. 6, pp. 38–52 (in Russian).
5. Boyarkina A. K., Ermolaeva V. V. Expert systems, *Molodoj uchenyi*, 2016, no. 11, pp. 286–289 (in Russian).
6. Eliseev D. V. Knowledge representation model at creation of the adaptive information system, *Nauka i obrazovanie: nauchnoe izdanie MGTU im. N. E. Baumana*, 2010, no. 3, pp. 1–6 (in Russian).
7. Bruynooghe M., Blockeel H., Bogaerts B. et al. Predicate logic as a modeling language: modeling and solving some machine learning and data mining problems with IDP3, *Theory and Practice of Logic Programming*, 2015, vol. 15, no. 6, pp. 783–817.
8. Strachova I. I. Model of knowledge representation and features of a logic model, *Sovremennye nauchnye issledovaniya i razrabotki*, 2017, vol. 2, no. 1, pp. 208–210 (in Russian).
9. Lavrishcheva E. M., Zelenov S. V., Pakulin N. V. Methods for evaluating the reliability of software and technical systems, *Trudy instituta sistemnogo programmirovaniya RAN*, 2019, vol. 31, no. 5, pp. 95–108 (in Russian).
10. Sidnyaev N. I. *Statistical analysis and theory of experiment planning*. Moscow, Bauman Moscow state technical University publishing house, 2017, 195 p. (in Russian).
11. Potyupkin A. Yu., Chechkin A. V. *Artificial intelligence on the basis of the information-system redundancy*, Moscow, KURS, 2019, 384 p. (in Russian).
12. Druzhinina O. V., Masina O. N. *Methods of stability analysis of dynamic systems of intelligent control*, Moscow, URSS, 2016. 248 p. (in Russian).
13. Vasenin V. A., Pirogov M. V., Chechkin A. V. *Information and system security of critical systems*, Moscow, KURS, 2018, 352 p. (in Russian).
14. Vasenin V. A., Pirogov M. V., Chechkin A. V. The Main Operator of Radical Modeling on a Demonstration Example, *Programmnaya Ingeneria*, 2016, vol. 7, no. 2, pp. 75–85 (in Russian).
15. Zybina E. Yu. About the identifiability of linear dynamic systems in closed loop in normal operation, *Izvestiya YUFU. Tekhnicheskie nauki*, 2015, no. 4 (165), pp. 160–170 (in Russian).
16. Sidnyaev N. I., Khrapov P. V. *Neural networks and neuro-mathematics*, Moscow, Bauman Moscow state technical University publishing house, 2016, 83 p. (in Russian).
17. Dyachenko R. A., Mahammad M. D. On the issue of evaluating the reliability of database management systems, *Vestnik IrGTU*, 2009, no. 1 (37), pp. 196–199 (in Russian).
18. Palyukh B. V., Burdo G. B. Knowledge Representation in computer-aided design and process control systems, *Vestnik TGTU*, 2010, no. 2, pp. 258–265 (in Russian).
19. Fridman A. Ya. Situational modeling of hierarchical dynamic systems, *Informacionnye i matematicheskie tekhnologii v naуke i upravlenii*, 2018, no. 1 (9), pp. 5–15 (in Russian).
20. Filimonov A. B., Filimonov N. B. Situational approach in tasks of automation of technical objects management, *Mekhanika, avtomatizaciya, upravlenie*, 2018, vol. 19, no. 9, pp. 563–578 (in Russian).

Д. Ю. Булгаков, адъюнкт, dbulgakov7@yandex.ru, Федеральное государственное казенное образовательное учреждение высшего образования "Академия управления Министерства внутренних дел Российской Федерации", Москва,
Я. В. Булгакова, канд. мед. наук, доц., yaroslava.v.bulgakova@mail.ru,
Н. А. Карапыгин, канд. биол. наук, доц., nikol.karatygin@yandex.ru,
Федеральное государственное автономное образовательное учреждение высшего образования "Первый Московский Государственный Медицинский Университет им. И. М. Сеченова", Москва

Современное свободное программное обеспечение для анализа и обработки электроэнцефалограмм: возможности и выбор

Проведен сравнительный анализ современных отечественных и зарубежных программных продуктов, находящихся в свободном доступе, для анализа и обработки электроэнцефалограмм (ЭЭГ) и других биомедицинских сигналов. Рассмотрены особенности использования самостоятельных приложений и специализированных библиотек, разработанных в популярных программных средах MATLAB, GNU Octave, Python. Материал может оказать помощь при выборе средств обработки ЭЭГ как начинающим исследователям, так и опытным специалистам.

Ключевые слова: свободное программное обеспечение, Python, цифровая обработка сигналов, ЦОС, биомедицинские сигналы, электроэнцефалография, NumPy, Brainstorm, MATLAB, EEGLAB

Введение

В настоящее время процессы, связанные с автоматизацией, охватывают многие сферы жизни: банковскую и финансовую деятельность, телекоммуникации, транспорт и др. [1]. Большое значение приобрела автоматизация цифровой обработки биомедицинских сигналов в практике научных исследований и клинической медицины. Так, при изучении процессов сознания, мышления, эмоциональной активности, интеллектуальной деятельности, а также диагностике ряда патологических состояний широко применяется регистрация биопотенциалов мозга — электроэнцефалография (ЭЭГ) [2–7]. Активно ведется разработка нейрокомпьютерных интерфейсов, в которых данные, полученные с помощью ЭЭГ, играют важную роль [7]. Вместе с тем выбор средств анализа и обработки ЭЭГ бывает весьма труден, поскольку проприетарное программное обеспечение (ПО), поставляемое с оборудованием, имеет ограниченные функциональные возможности, что является проблемой, прежде всего, для научных исследований, а обилие различного рода свободного ПО¹ нередко требует узкоспециальных навыков при

обращении с ним. При этом необходимо отметить, что сопоставление возможностей современных программных продуктов, используемых для обработки ЭЭГ, встречается в крайне ограниченном числе литературных источников [4, 8].

В связи с изложенными выше обстоятельствами целью настоящей работы явились сравнительный анализ свободного ПО с открытым исходным кодом и систематизация данных по вопросам выбора формата обработки и хранения ЭЭГ.

лицензии, которые позволяют пользователю: использовать программу для ЭВМ в любых, не запрещенных законом целях; получать доступ к исходным текстам (кодам) программы как в целях изучения и адаптации, так и в целях переработки программы для ЭВМ; распространять программу (бесплатно или за плату, по своему усмотрению); вносить изменения в программу для ЭВМ (перерабатывать) и распространять экземпляры измененной (переработанной) программы с учетом возможных требований наследования лицензии; в отдельных случаях распространять модифицированную компьютерную программу пользователем на условиях, идентичных тем, на которых ему предоставлена исходная программа. **Проприетарное (закрытое) программное обеспечение:** Программное обеспечение, распространяемое на условиях простой (неисключительной) или исключительной лицензии, ограничивающей использование программы и/или запрещающей пользователю внесение изменений в программу для ЭВМ (переработку) и/или распространение изменений (переработанной) программы».

¹ В соответствии с ГОСТ Р 54593–2011 «Свободное программное обеспечение. Общие положения». «Свободное программное обеспечение»: Программное обеспечение, распространяемое на условиях простой (неисключительной) ли-

Форматы файлов, используемые для хранения ЭЭГ-сигналов

Анализ источников [9–11] показывает, что в настоящее время существует большое число различных форматов для хранения записей ЭЭГ, однако во многих коммерческих продуктах и в исследовательских проектах используется формат EDF¹ [9], который является международным открытым стандартом хранения многоканальных биологических и физических сигналов [10]. В 2002 г. формат EDF был доработан: были добавлены возможность сохранения аннотаций, информации о стимуляциях и событиях, устранины некоторые недостатки EDF-формата, такие как "проблема 2000 года", и др. Новый формат получил название EDF+ [10, 11]. Форматы совместимы — существующие средства просмотра корректно работают с файлами обоих типов. Необходимо учитывать, что стандарт EDF+ для описания хранимых данных предписывает использовать только латинские буквы, цифры, знаки препинания и некоторые специальные символы, поэтому использовать кириллицу внутри файлов EDF+ не рекомендуется.

В ряде случаев используется версия формата с большей разрядностью — EDF32. Реже можно встретиться с данными ЭЭГ, записанными в файлах формата BioSemi, имеющих расширение BDF. Этот формат (BDF/EDF+) является расширением форматов EDF/EDF+, при этом уровень квантования (разрядность) записываемого сигнала составляет 24 бита, в то время как в EDF/EDF+ этот показатель равен 16 битам [10, 11].

Также в целях преодоления некоторых ограничений формата EDF был разработан формат GDF². Этот 24/32-битный универсальный формат позволяет хранить дополнительную информацию о субъекте исследования, подробную информацию о каждом отведении и обеспечивает расширенный набор функций по сравнению с другими форматами [7].

Многие программы записи и обработки ЭЭГ позволяют сохранять данные в виде обычного текстового файла (plain text ASCII). Это дает возможность считывать данные без привлечения специализированных библиотек.

Таким образом, в большинстве случаев наиболее простым и удобным форматом хранения данных ЭЭГ можно считать EDF+, который является "де факто" стандартом для хранения записей ЭЭГ и поддерживается большинством приложений. Если же требуется сохранить большой объем дополнительных данных, используются более сложные форматы, например, BDF или GDF.

Просмотр ЭЭГ-файлов

Существуют специализированные программы, предназначенные в первую очередь для просмотра

¹ EDF — от англ. European Data Format — европейский формат данных.

² GDF — от англ. General Data Format for Biomedical Signals — общий формат данных для биомедицинских сигналов.

и аннотирования ЭЭГ. К ним можно отнести EDF-browser, EDF Read, SigViewer, Polymax [12–14]. Данное ПО является свободным.

Для просмотра и анализа ЭЭГ существует большое число коммерческих программ с закрытым исходным кодом, поставляемых с энцефалографом или приобретаемых отдельно. В них уже заложены определенные алгоритмы обработки электроэнцефалограмм, модифицировать которые пользователь не имеет возможности. Поэтому в данной работе внимание сосредоточено на характеристиках свободного ПО.

Языковые платформы MATLAB и Python

Анализ литературных источников показывает, что для обработки ЭЭГ-сигналов исследователи часто применяют математический программный комплекс MATLAB [15]. Для разработки модулей в MATLAB используется одноименный язык программирования, ориентированный в большей степени не на профессиональных программистов, а на ученых-исследователей (табл. 1). В его состав входят модули для базовой обработки сигналов, а также матриц и временных рядов, в том числе функции для работы с преобразованиями Фурье и вейвлетными преобразованиями. Модули для расширенного анализа сигналов включают проектирование и разработку сложных систем цифровой обработки, библиотеки классических вейвлетов (Добеши, комплексные вейвлеты Гаусса и Морле, дискретный Мейера и др.), средства разработки вейвлет-алгоритмов.

С 1996 г. существует и активно развивается свободный аналог MATLAB — программный пакет GNU Octave [16] (см. табл. 1). Для него создано большое число свободных специализированных модулей для решения различных задач [17].

Язык программирования Python распространяется под свободной лицензией, имеет большое число модулей расширения и представляет в области обработки сигналов достойную альтернативу MATLAB [18]. С помощью специализированного модуля NumPy в Python имеется возможность работы с многомерными массивами [19], в виде которых обычно представлены данные биосигналов, создано большое число качественных, быстродействующих расширений для обработки и анализа научных данных (SciPy, NumExpr, Pandas), а также визуализации получаемых результатов (Matplotlib, Bokeh) [19, 20]. Семейство алгоритмов преобразования Фурье входит в состав пакетов NumPy и SciPy. Для научных вычислений удобен кросс-платформенный бесплатный дистрибутив Anaconda Python, в состав которого уже включены необходимые пакеты расширения — NumPy, скомпилированный с поддержкой высокопроизводительной библиотеки Intel MKL³, SciPy, Pandas, Matplotlib, Bokeh и др. Алго-

³ Intel Math Kernel Library — библиотека оптимизированных математических программ для научных, инженерных и финансовых приложений. Основные математические функции значительно ускоряют вычисления в библиотеках BLAS, LAPACK, ScaLAPACK, выполнение операций быстрого преобразования Фурье и векторной математики.

Таблица 1

Базовое ПО, на основе которого работают программы/библиотеки для обработки ЭЭГ

Название ПО	Официальный сайт	Характеристика	Тип	Поддерживаемые операционные системы
MATLAB	https://www.mathworks.com/products/matlab.html	Специализированный математический комплекс с языком программирования MATLAB (платное ПО)	Проприетарное. Самостоятельное приложение	Windows, Linux, macOS
MATLAB Runtime	https://www.mathworks.com/products/compiler/matlab-runtime.html	Библиотеки для запуска ПО, написанного на MATLAB (бесплатное ПО)	Проприетарное. Предназначено для запуска других программ	
GNU Octave	https://www.gnu.org/software/octave/	Специализированный математический комплекс с языком программирования, совместимым с MATLAB	Свободное. Самостоятельное приложение	Windows, Linux, macOS, FreeBSD, OpenBSD
Python	http://www.python.org	Язык программирования (интерпретатор программ, написанных на языке Python)	Свободное. Самостоятельное приложение	Windows, Linux, macOS + большое число операционных систем и аппаратных платформ

ритмы для работы с вейвлет-преобразованием и наиболее распространенные вейвлет-функции (свыше 100) входят в состав пакета PyWavelets.

Необходимо отметить, что Python и MATLAB хорошо взаимодействуют. Из программ, написанных на Python, можно вызывать программы, написанные на MATLAB, и наоборот [21, 22].

Свободное ПО для анализа ЭЭГ

Рассмотрим наиболее крупные программные комплексы, распространяемые как свободное ПО (табл. 2).

Команда исследователей из Нидерландов активно развивает свободное дополнение к MATLAB — модуль FieldTrip toolbox [8, 23]. Данный модуль представляет собой библиотеку с большим набором функций языка MATLAB. Некоторые из них основаны на дополнительных модулях Image Processing, Optimization, Signal Processing, Statistics, приобретаемых за отдельную плату [23]. Развитие модуля FieldTrip toolbox направлено на уход от заимствований функциональных возможностей MATLAB и самостоятельную реализацию решений на базе GNU Octave (см. табл. 2).

Для MATLAB разработан специализированный свободный модуль EEGLAB [24]. Помимо задачи чтения данных из внешних файлов, модуль предоставляет возможности обработки сигналов методом анализа независимых компонент¹, методом частотно-временного анализа², позволяет проводить статистический анализ локальных феноменов, имеет несколь-

ко режимов визуализации усредненных данных по нескольким исследованиям и данных единичных испытаний, включает средства удаления артефактов³.

На основе MATLAB создан мощный программный комплекс с открытым исходным кодом Brainstorm, объединяющий большое число пользователей из разных стран и сопоставимый по своим возможностям с коммерческими программными продуктами. Комплекс предназначен для анализа МЭГ, ЭЭГ, электрокортикограмм и ЭЭГ животных [25]. Ежегодно публикуется большое число статей, посвященных работе с Brainstorm, проводятся семинары и учебные курсы, выпускаются проблемно-ориентированные учебные материалы [25]. При отсутствии лицензии на MATLAB понадобится специальная версия Brainstorm — Alternate binary distribution и бесплатный набор Runtime-компонентов MATLAB Compiler Runtime, необходимый для запуска приложений, написанных для MATLAB [26].

На языке Python написан программный комплекс, предназначенный для визуального и полуавтоматического анализа ЭЭГ Visbrain, который позволяет, в числе прочего, строить топокарты, выявлять функциональные связи между различными участками головного мозга [27, 28].

Специалистами институтов США и Франции разработан мощный программный комплекс с открытым исходным кодом MNE, предназначенный для анализа ЭЭГ, магнитоэнцефалограмм (МЭГ) и электрокортикограмм. Разработка MNE ведется на нескольких языках программирования [29, 30].

В настоящее время проект MNE представляет собой большой многоцелевой программный ком-

¹ Independent Component Analysis (ICA).

² Time/frequency analysis.

³ Artifact rejection.

Таблица 2

Характеристики программ/библиотек для обработки ЭЭГ с открытым исходным кодом

Название	Официальный сайт	Характеристика	Для работы необходим	Интерфейс пользователя	Поддерживаемые операционные системы
Brainstorm	http://neuroimage.usc.edu/brainstorm/	Законченное приложение для анализа ЭЭГ. Простое в установке, имеет мощные средства обработки, анализа и визуализации ЭЭГ. Сложная для пользователя работа в приложении	MATLAB, MATLAB Runtime	Есть	Windows, Linux, macOS
EEGLAB	https://sccn.ucsd.edu/~scott/ica.html	Законченное приложение для анализа ЭЭГ, МЭГ	MATLAB, GNU Octave*	Есть	
MNE-Python	http://martinos.org/mne	Библиотека Python для анализа ЭЭГ и работы с EDF-файлами	Python	Нет	
The BioSig Project	http://biosig.sourceforge.net	Библиотека для анализа ЭЭГ	MATLAB, GNU Octave	Нет	
HAPPE	https://www.frontiersin.org/articles/10.3389/fnins.2018.00097/full	Библиотека для анализа ЭЭГ	MATLAB версии 2014 и выше	Нет	
FieldTrip toolbox	http://www.fieldtriptoolbox.org	Библиотека для анализа ЭЭГ, МЭГ	MATLAB, GNU Octave	Нет	
Visbrain	http://visbrain.org	Библиотеки + приложение. Запуск требует квалифицированного пользователя	Python	Есть	
MNE-CPP	http://martinos.org/mne	Приложение для обработки и визуализации ЭЭГ	Нет требований	Есть	
MathBrain	http://www.mathbrain.ru	Интернет-ресурс для анализа данных ЭЭГ, допуск по модели "Приложение как сервис"	Интернет	Есть	
SigViewer	https://github.com/cbrnr/sigviewer	Приложение для просмотра сигналов в EDF/BDF-файлах	Нет требований	Есть	
EDFbrowser	https://www.teuniz.net/edfbrowser/	Приложение для просмотра, разметки (аннотирования) и проверки корректности формата файлов с записями ЭЭГ-сигналов	Нет требований	Есть	
Polyman	https://www.edfplus.info/downloads/	Приложение для просмотра сигналов в файлах EDF/EDF+, просмотра видеоданных и работы с аннотациями	Нет требований	Есть	Windows

* – Работоспособность под GNU Octave не гарантируется.

плекс, состоящий из нескольких модулей. Ядро комплекса написано на языке C (модуль MNE-C), для взаимодействия с пользовательскими программами на Python используется модуль MNE-Python, на развитии которого сейчас сосредоточено основное внимание разработчиков MNE. Для взаимодействия с программами на MATLAB используется модуль MNE-MATLAB toolbox. Модуль MNE-CPP написан на языке C++ и используется для получения в реальном времени данных ЭЭГ, их анализа и визуализации [31].

Задачи, решение которых возможно с использованием MNE, весьма разнообразны: работа с заранее подготовленными наборами данных — датасетами, предобработка данных — использование различных

фильтров, распознавание и коррекция артефактов, анализ независимых компонентов, фильтрация Максвелла и др., визуализация данных.

Проект MNE позволяет реализовать машинное обучение. Если база собственных экспериментальных данных невелика, то для машинного обучения, проверки гипотез, создания статистических моделей создано большое количество датасетов (наборов данных) с записанными и классифицированными ЭЭГ-сигналами. Они сгруппированы по определенным тематикам: выявление и удаление шумов, работа с артефактами, распознавание эмоций, диагностика заболеваний [32]. Датасет DEAP, содержащий ЭЭГ эмоциональных проявлений, включает видеозапись лиц испытуемых. Данные материалы находятся

в свободном доступе. Записи представлены в форматах EDF, BDF, MATLAB, pru/prz¹, что позволяет значительно ускорить их загрузку с диска в память и обработку в соответствующих специализированных программах [34].

Взаимодействие с компонентами комплекса MNE предусмотрено в упомянутых выше проектах Brainstorm и Visbrain.

Библиотека The Large Time-Frequency Analysis Toolbox (LTFAT) может быть использована для проведения частотно-временного анализа, синтеза и преобразования сигналов [35]. В ней, в числе прочего, содержится 54 функции для осуществления преобразования Фурье и косинусного дискретного преобразования, а также 70 различных функций для работы с вейвлетами [36]. В то же время в LTFAT отсутствуют функции для загрузки сведений из EDF-файлов, а также файлов других форматов, используемых для хранения биомедицинских сигналов. Для этого рекомендуется использовать свободную библиотеку The BioSig Project [11, 37]. BioSig поддерживает работу с файлами форматов EDF/EDF+, BDF/BDF+ и некоторыми форматами файлов, которые используются в коммерческом ПО, поставляемом с ЭЭГ-оборудованием, содержит модули для визуализации и проведения специфических вычислений [11].

Для автоматизации анализа ЭЭГ при проведении крупных исследований целесообразно использовать свободную библиотеку The Harvard Automated Processing Pipeline for Electroencephalography (HAPPE), которая позволяет проводить комплексную обработку ЭЭГ-сигналов [39]. Для работы HAPPE необходим MATLAB с модулями-расширениями Signal processing toolbox, Optimization toolbox, Statistics toolbox.

В продолжение развития HAPPE создана библиотека The Batch Electroencephalography Automated Processing Platform (BEAPP) для автоматизированной пакетной обработки ЭЭГ [40]. BEAPP объединяет программные наработки других инструментов анализа ЭЭГ, включая HAPPE, EEGLAB, FieldTrip, MARA² и др., в результате чего пользователь может в рамках одной библиотеки получить доступ к наиболее востребованным функциональным возможностям этих продуктов. BEAPP позволяет загрузить сырье или прошедшие первичную обработку данные и затем в автоматическом режиме провести удаление артефактов, а также ряд других корректирующих процедур, осуществить спектральный, когерентный анализ, анализ вызванных потенциалов и др. BEAPP предоставляет собственный интерфейс и не требует от пользователя написания программного кода.

В соответствии с современными тенденциями представления ПО как сервиса, Российский Институт математических проблем биологии разрабатывает проект MathBrain [41]. Ресурс не требует установки специализированного ПО и может быть использован

для обработки данных ЭЭГ только с использованием браузера. Набор инструментов MathBrain дает возможность работы с полными временными рядами, со скользящим окном, восстановления временных рядов по рассчитанному спектру, позволяет осуществить преобразование Фурье, анализ главных компонент, решение обратной задачи магнитоэнцефалографии на основе многоканальных спектральных данных. С технической точки зрения роль основного вычислительного механизма в программном комплексе выполняет набор скриптов Python, использующих в своей работе упомянутые выше библиотеки NumPy, SciPy и NumExpr.

Заключение

Количество программного обеспечения, предназначенного для обработки и анализа ЭЭГ, в том числе с открытым исходным кодом, в настоящее время весьма велико. Поэтому уже на этапе планирования исследования его выбору должно бытьделено достаточное внимание. Важным является определение методов анализа, которые предполагается использовать при обработке полученных данных. Если для целей исследования достаточно комплекса традиционных методов, вероятно, оптимальным выбором будет использование проприетарного ПО, идущего в комплекте с энцефалографом (если таковое имеется). Можно использовать и свободное ПО: например, программный комплекс Brainstorm имеет достаточно широкие возможности и позволяет работать с сигналами ЭЭГ пользователям, не являющимся программистами или математиками.

Если требуется специфическая обработка данных ЭЭГ, то можно использовать многочисленные библиотеки или отдельные приложения на базе таких языковых сред, как MATLAB (или ее свободного аналога GNU Octave), Python. Для среды MATLAB можно порекомендовать такие продукты, как EEGLAB, The BioSig Project, FieldTrip toolbox. Для Python возможно использование программных комплексов MNE-Python и Visbrain. При этом следует быть готовым самостоятельно писать часть программного кода для автоматизации взаимодействия различных библиотек и преобразования файлов ЭЭГ.

Планируя обработку данных ЭЭГ в целях проведения научных исследований, рекомендуется предварительно изучить возможности проприетарного ПО, поставляемого в комплекте с используемым оборудованием, на предмет того, в каком формате это ПО способно сохранять (экспортировать) данные ЭЭГ. Если сохранить ЭЭГ в распространенных открытых форматах, то дальнейшую обработку исходных данных можно выстроить на базе распространенного ПО, в том числе с открытым исходным кодом, доступного другим исследователям. Полученное решение будет отчуждаемым³, тиражируемым и воспроизводимым. Исходные данные ЭЭГ можно оформить как датасет, опубликовать с присвоением

¹ pru, prz — формат сохранения информации из многомерных массивов Python NumPy во внешний файл на диске: pru — данные хранятся в несжатом виде; prz — данные хранятся в сжатом виде.

² MARA — Multiple Artifact Rejection Algorithm.

³ Независимым от конкретного оборудования и ПО, поставляемого с ним в комплекте.

DOI [42] и зарегистрировать в Роспатенте в качестве базы данных [43].

В целом, можно констатировать, что обработка сигналов ЭЭГ является динамичной и быстро развивающейся областью знаний, в которой продолжают возникать новые подходы, методы и алгоритмы анализа. Большинство рассмотренных программных комплексов активно развивается, поддерживается разработчиками и позволяет работать со все большим числом сложных методов анализа ЭЭГ, МЭГ, МРТ и др. Существенным моментом является то, что использование свободного ПО позволяет самостоятельно и индивидуально подойти к выбору и, зачастую, созданию средств обработки сигнала, при этом позволяет пользователю быть интегрированным в мировое научное сообщество, что весьма важно при публикациях в зарубежных изданиях. Также безусловным и очевидным преимуществом свободного ПО является отсутствие финансовых затрат на использование и поддержку. Принимая во внимание приведенные сведения, у исследователя есть возможность подобрать наиболее актуальный программный продукт, максимально соответствующий задачам конкретного исследования.

Исследование выполнено при поддержке гранта РФФИ № 19-29-01156 МК.

Список литературы

1. Позин Б. А. Проблемы программной инженерии на современном этапе ее развития // Программная инженерия. — 2011. — № 1. — С. 2–6.
2. Shestyuk A. Y., Kasinathan A. Y., Karthik K. et al. Individual EEG measures of attention, memory, and motivation predict population level TV viewership and Twitter engagement // PLoS One. — 2019. — Vol. 14, No. 3. — P. 1–27.
3. Beekoo D., Yuan K., Dai S. et al. Analyzing electroencephalography (EEG) waves provides a reliable tool to assess the depth of sevoflurane anesthesia in pediatric patients // Med. Sci. Monit. — 2019. — Vol. 25. — P. 4035–4040.
4. Smailovic U., Jelic V. Neurophysiological Markers of Alzheimer's Disease: Quantitative EEG Approach // Neurol. Ther. Adis. — 2019. — Vol. 8, No. S2. — P. 37–55.
5. Obeid I., Picone J. The Temple University Hospital EEG Data Corpus // Front. Neurosci. Frontiers Media S. A. — 2016. — Vol. 10, No. MAY. — P. 196.
6. Athavipach C., Panngum S., Israsena P. A Wearable In-Ear EEG Device for Emotion Monitoring // Sensors. MDPI AG. — 2019. — Vol. 19, No. 18. — P. 4014.
7. Туровский Я. А., Суровцев А. С., Зайцев С. А. Коновской А. С. Переходные процессы зрительных вызванных потенциалов в задачах интерфейсов человек-компьютер // Сенсорные системы. — 2019. — Т. 33, № 2. — С. 157–165.
8. Oostenveld R., Fries P., Maris E., Schoffelen J. M. FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data // Comput. Intell. Neurosci. Hindawi Limited. — 2011. — Vol. 2011. URL: <http://www.fieldtriptoolbox.org/> (дата доступа 26.03.2020).
9. Kemp B., Värrí A., Rosa A. C. et al. A simple format for exchange of digitized polygraphic recordings // Electroencephalogr. Clin. Neurophysiol. — 1992. — Vol. 82, No. 5. — P. 391–393.
10. Kemp B., Olivan J. European data format ‘plus’ (EDF+), an EDF alike standard format for the exchange of physiological data // Clin. Neurophysiol. Elsevier Ireland Ltd. — 2003. — Vol. 114, No. 9. — P. 1755–1761.
11. Schlogl A. An overview on data formats for biomedical signals/ Eds Dössel O., Schlegel W. C. // World Congress on Medical Physics and Biomedical Engineering, September 7–12, 2009, Munich, Germany. IFMBE Proceedings, vol. 25/4. Springer, Berlin. — 2009. — P. 1557–1560.
12. Github. Sigviewer. URL: <https://github.com/cbrnr/sigviewe> (дата доступа 24.03.2020).
13. Alvarez-Estevez D. Polyman. URL: <https://sites.google.com/view/diegoalvarezestevez/projects/polyman> (дата доступа 24.03.2020).
14. EDFbrowser. URL: <https://www.teuniz.net/edfbrowser/> (дата доступа 24.03.2020).
15. MATLAB — MathWorks — MATLAB & Simulink. URL: <https://www.mathworks.com/products/matlab.html> (дата доступа 24.03.2020).
16. GNU Octave [Electronic resource]. URL: <https://www.gnu.org/software/octave/> (дата доступа 24.03.2020).
17. Octave Forge — Packages. URL: <https://octave.sourceforge.io/packages.php> (дата доступа 24.03.2020).
18. Welcome to Python. URL: <https://www.python.org/> (дата доступа 24.03.2020).
19. NumPy — NumPy. URL: <https://numpy.org/> (дата доступа 24.03.2020).
20. NumPy for Matlab users — NumPy v1.14 Manual. URL: <https://docs.scipy.org/doc/numpy-1.14.2/user/numpy-for-matlab-users.html> (дата доступа 24.03.2020).
21. Lee G., Gommers R., Waselewski F. et al. PyWavelets: A Python package for wavelet analysis // J. Open Source Softw. The Open Journal. — 2019. — Vol. 4, No. 36. — P. 1237.
22. Using MATLAB with Python — MATLAB & Simulink. URL: <https://www.mathworks.com/products/matlab/matlab-and-python.html> (дата доступа 26.03.2020).
23. Download the FieldTrip toolbox — FieldTrip toolbox. URL: http://www.fieldtriptoolbox.org/download/#what_are_the_matlab_requirements_for_using_fieldtrip (дата доступа 24.03.2020).
24. Delorme A., Makeig S. EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including independent component analysis // J. Neurosci. Methods. — 2004. — Vol. 134, No. 1. — P. 9–21.
25. Tadel F., Bock E., Niso G. et al. MEG/EEG group analysis with brainstorm // Front. Neurosci. — 2019. — Vol. 13, No. FEB. — P. 1–21.
26. MATLAB Runtime — MATLAB Compiler — MATLAB. URL: <https://www.mathworks.com/products/compiler/matlab-runtime.html> (дата доступа 24.03.2020).
27. Contents — visbrain 0.4.6 documentation. URL: <http://visbrain.org/> (дата доступа: 24.03.2020).
28. Combrisson E., Vallat R., O'Reilly C. et al. Visbrain: A Multi-Purpose GPU-Accelerated Open-Source Suite for Multimodal Brain Data Visualization // Front. Neuroinform. — 2019. — Vol. 13. — P. 14.
29. Gramfort A., Luessi M., Larson E. et al. MNE software for processing MEG and EEG data // Neuroimage. — 2014. — Vol. 86. — P. 446–460.
30. Publications by users — MNE 0.19.2 documentation. URL: <https://mne.tools/stable/cited.html> (дата доступа 24.03.2020).
31. MNE-CPP — MNE-CPP: A Framework for Electrophysiology. URL: <https://www.mne-cpp.org/> (дата доступа 24.03.2020).
32. Free EEG data database freely ERP data publicly available. URL: https://sccn.ucsd.edu/~arno/fam2data/publicly_available_EEG_data.html (дата доступа 24.03.2020).
33. PhysioNet Index. URL: <https://www.physionet.org/data/#neur> (дата доступа 24.03.2020).

-
34. **Numpy.load** — NumPy v1.17 Manual. URL: <https://docs.scipy.org/doc/numpy/reference/generated/numpy.load.html> (дата доступа 24.03.2020).
35. **The Large** Time-Frequency Analysis Toolbox. URL: <http://ltfat.github.io/> (дата доступа: 24.03.2020).
36. **LTFAT** — Wavelets. URL: <http://ltfat.github.io/doc/wavelets/> (дата доступа 24.03.2020).
37. **BioSig**. URL: <http://biosig.sourceforge.net/> (дата доступа 24.03.2020).
38. **Biosig**. URL: <http://pub.ist.ac.at/~schloegl/biosig/TESTED> (дата доступа 24.03.2020).
39. **Gabard-Durnam L. J., Mendez L. A., Wilkinson C. L., Levin A. R.** The Harvard Automated Processing Pipeline for Electroencephalography (HAPPE): Standardized Processing Software for Developmental and High-Artifact Data // Front. Neurosci. — 2018. — Vol. 12. — P. 97.
40. **Levin A. R., Méndez Leal A. S., Gabard-Durnam L. J., O'Leary H. M.** BEAPP: The Batch Electroencephalography Automated Processing Platform // Front. Neurosci. Frontiers Media S. A. — 2018. — Vol. 12, No. 513. — P. 513.
41. **Оплачко Е. С., Рыкунов С. Д., Устинин М. Н.** Облачный ресурс MathBrain для обработки данных энцефалографии // Препринты ИПМ им. М. В. Келдыша. — 2018. — № 10. — 20 с. DOI:10.2094. URL: http://keldysh.ru/papers/2018/prep2018_10.pdf (дата доступа 24.03.2020).
42. **Share** and discover datasets. URL: <https://www.mendeley.com/datasets> (дата доступа 19.05.2020).
43. **Программы для ЭВМ, БД.** URL: <https://new.fips.ru/to-applicants/software-and-databases/> (дата доступа 19.05.2020).
-

Up-to-date Open-Source Software for the Analysis and Processing of Electroencephalograms: Opportunities and the Choices

D. Yu. Bulgakov, dbulgakov7@yandex.ru, Management Academy of the Ministry of the Interior of Russia, Moscow, 125993, Russian federation, **Ya. V. Bulgakova**, yaroslava.v.bulgakova@mail.ru, **N. A. Karatygin**, nikol.karatygin@yandex.ru, I. M. Sechenov First Moscow State Medical University (Sechenov University), Moscow, 119435, Russian Federation

Corresponding author:

Bulgakova Yaroslava V., Associate Professor, I. M. Sechenov First Moscow State Medical University (Sechenov University), 119435, Moscow, Russian Federation
E-mail: yaroslava.v.bulgakova@mail.ru

*Received on March 30, 2020
Accepted on June 09, 2020*

The authors carried out the analytic comparison of up-to-date open-source software for the analysis and processing of electroencephalograms (EEG) and other biomedical signals. We consider the features of stand-alone applications and specialized libraries developed in the popular software environments MATLAB, GNU Octave, Python. Materials include a comparative description of the software for biomedical signals processing.

Authors conclude that if traditional methods are sufficient for the purposes of the study, it would be optimal to use the proprietary software that comes with the equipment. But an investigator can use open-source software as well. For example, the "Brainstorm" software has quite wide capabilities and allows users who are not programmers or mathematicians to work with EEG signals.

If specific processing of EEG data is required, then you can use software libraries from such software systems as MATLAB (or its free analogue GNU Octave) or Python. For MATLAB, products like EEGLAB, The BioSig Project, FieldTrip toolbox can be recommended. For Python, you can use MNE-Python and Visbrain. In this case, you should be ready to write a part of the program code yourself to automate the interaction of various libraries and convert EEG files.

The material assists in the selection of EEG processing tools for both beginners and experienced professionals.

Keywords: open-source software, Python, digital signal processing, DSP, biomedical signals, electroencephalography, NumPy, Brainstorm, MATLAB, EEGLAB

Acknowledgements: This work was supported by the Russian Foundation for Basic Research, project No. 19-29-01156 MK.

For citation:

Bulgakov D. Yu., Bulgakova Ya. V., Karatygin N. A. Up-to-date Open-Source Software for the Analysis and Processing of Electroencephalograms: Opportunities and the Choices, *Программная Инженерия*, 2020, vol. 11, no. 4, pp. 205–212

DOI: 10.17587/prin.11.205-212

References

1. **Posin B. A.** Problems of software engineering at the present stage of its development, *Programmnaya Ingeneria*, 2011, no. 1, pp. 2–6. (in Russian).
2. **Shestyuk A. Y., Kasinathan A. Y., Karthik K.** et al. Individual EEG measures of attention, memory, and motivation predict population level TV viewership and Twitter engagement, *PLoS One*, 2019, vol. 14, no. 3, pp. 1–27.
3. **Beekoo D., Yuan K., Dai S.** et al. Analyzing electroencephalography (EEG) waves provides a reliable tool to assess the depth of sevoflurane anesthesia in pediatric patients, *Med. Sci. Monit*, 2019, vol. 25, pp. 4035–4040.
4. **Smailovic U., Jelic V.** Neurophysiological Markers of Alzheimer's Disease: Quantitative EEG Approach, *Neurol. Ther. Adis*, 2019, vol. 8, no. S2, pp. 37–55.
5. **Obeid I., Picone J.** The Temple University Hospital EEG Data Corpus *Front. Neurosci.*, 2016, vol. 10, no. 5, pp. 196.
6. **Athavipach C., Panngum S., Israsena P.** A Wearable In-Ear EEG Device for Emotion Monitoring, *Sensors*, 2019, vol. 19, no. 18, pp. 4014.
7. **Turovsky Y. A., Surovtsev A. S., Zaitsev S. A., Konovskoy A. S.** Transient processes of visual evoked potentials in the tasks of human-computer interfaces *Sensory systems*, 2019, vol. 33, no. 2, pp. 157–165.
8. **Oostenveld R., Fries P., Maris E., Schoffelen J. M.** FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data, *Comput. Intell. Neurosci.*, 2011, vol. 2011, available at: <http://www.fieldtriptoolbox.org/> (date of access 28.03.2020).
9. **Kemp B., Värrä A., Rosa A. C.** et al. A simple format for exchange of digitized polygraphic recordings, *Electroencephalogr. Clin. Neurophysiol.*, 1992, vol. 82, no. 5, pp. 391–393.
10. **Kemp B., Olivan J.** European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data, *Clin. Neurophysiol.*, 2003, vol. 114, no. 9, pp. 1755–1761.
11. **Schlögl A.** An overview on data formats for biomedical signals, *IFMBE Proceedings*, 2009, vol. 12, no. 25/4, pp. 1557–1560.
12. **Github.** Sigviewer, available at: <https://github.com/cbrnr/sigviewe> (date of access 24.03.2020).
13. **Diego Alvarez-Estevez** — Polyman, available at: <https://sites.google.com/view/diegoalvarezestevez/projects/polyman> (date of access 24.03.2020).
14. **EDFbrowser**, available at: <https://www.teuniz.net/edfbrowser/> (date of access 24.03.2020).
15. **MATLAB** — MathWorks — MATLAB & Simulink, available at: <https://www.mathworks.com/products/matlab.html> (date of access 24.03.2020).
16. **GNU Octave**, available at: <https://www.gnu.org/software/octave/> (date of access 24.03.2020).
17. **Octave Forge** — Packages, available at: <https://octave.sourceforge.io/packages.php> (date of access 24.03.2020).
18. **Welcome** to Python, available at: <https://www.python.org/> (date of access 24.03.2020).
19. **NumPy** — NumPy, available at: <https://numpy.org/> (date of access 24.03.2020).
20. **NumPy for Matlab users** — NumPy v1.14 Manual, available at: <https://docs.scipy.org/doc/numpy-1.14.2/user/numpy-for-matlab-users.html> (date of access 24.03.2020).
21. **Lee G., Gommers R., Waselewski F.** et al. PyWavelets: A Python package for wavelet analysis, *J. Open Source Soft*, 2019, vol. 4, no. 36, p. 1237.
22. **Using MATLAB with Python** — MATLAB & Simulink, available at: <https://www.mathworks.com/products/matlab/matlab-and-python.html> (date of access 26.03.2020).
23. **Download the FieldTrip toolbox** — FieldTrip toolbox, available at: http://www.fieldtriptoolbox.org/download/#what_are_the_matlab_requirements_for_using_fieldtrip (date of access 24.03.2020).
24. **Delorme A., Makeig S.** EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including independent component analysis, *J. Neurosci. Methods*, 2004, vol. 134, no. 1, pp. 9–21.
25. **Tadel F., Bock E., Niso G.** et al. MEG/EEG group analysis with brainstorm, *Front. Neurosci.* 2019, vol. 13, no. FEB, pp. 1–21.
26. **MATLAB Runtime** — MATLAB Compiler — MATLAB, available at: <https://www.mathworks.com/products/compiler/matlab-runtime.html> (date of access 24.03.2020).
27. **Contents** — visbrain 0.4.6 documentation, available at: <http://visbrain.org/> (date of access 24.03.2020).
28. **Combrisson E., Vallat R., O'Reilly C.** et al. Visbrain: A Multi-Purpose GPU-Accelerated Open-Source Suite for Multimodal Brain Data Visualization, *Front. Neuroinform.*, 2019, vol. 13, p. 14.
29. **Gramfort A., Luessi M., Larson E.** et al. MNE software for processing MEG and EEG data, *NeuroImage*, 2014, vol. 86, pp. 446–460.
30. **MNE 0.19.2** documentation, available at: <https://mne.tools/stable/cited.html> (date of access 24.03.2020).
31. **MNE-CPP** — MNE-CPP: A Framework for Electrophysiology, available at: <https://www.mne-cpp.org/> (date of access 24.03.2020).
32. **Free EEG** data database freely ERP data publicly available, available at: https://sccn.ucsd.edu/~arno/fam2data/publicly_available_EEG_data.html (date of access 24.03.2020).
33. **PhysioNet** Index, available at: <https://www.physionet.org/data/#neur> (date of access 24.03.2020).
34. **Numpy.load** — NumPy v1.17 Manual, available at: <https://docs.scipy.org/doc/numpy/reference/generated/numpy.load.html> (date of access 24.03.2020).
35. **The Large** Time-Frequency Analysis Toolbox, available at: <http://ltfat.github.io/> (date of access 24.03.2020).
36. **LTFAT** — Wavelets, available at: <http://ltfat.github.io/doc/wavelets/> (date of access 24.03.2020).
37. **BioSig**, available at: <http://biosig.sourceforge.net/> (date of access 24.03.2020).
38. **Biosig**, available at: <http://pub.ist.ac.at/~schloegl/biosig/TESTED> (date of access 24.03.2020).
39. **Gabard-Durnam L. J., Mendez L. A., Wilkinson C. L., Levin A. R.** The Harvard Automated Processing Pipeline for Electroencephalography (HAPPE): Standardized Processing Software for Developmental and High-Artifact Data, *Front. Neurosci.*, 2018, vol. 12, p. 97.
40. **Levin A. R., Méndez Leal A. S., Gabard-Durnam L. J., O'Leary H. M.** BEAPP: The Batch Electroencephalography Automated Processing Platform, *Front. Neurosci.*, 2018, vol. 12, no. 513, p. 513.
41. **Oplachko E. S., Rykunov S. D., Ustinin M. N.** MathBrain Cloud Resource for Encephalography Processing, *M. V. Keldysh KIAM Preprints*, 2018, no. 10, 20 p., available at: http://keldysh.ru/papers/2018/prep2018_10.pdf (date of access 24.03.2020) (in Russian).
42. **Share and discover datasets**, available at: <https://www.mendeley.com/datasets> (date of access 19.05.2020).
43. **FIPS**, available at: <https://new.fips.ru/to-applicants/software-and-databases/> (date of access 19.05.2020).

А. В. Мальцев, канд. физ.-мат. наук, вед. науч. сотр., avmaltcev@mail.ru,
Федеральное государственное учреждение "Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук"
(ФГУ ФНЦ НИИСИ РАН), Москва

Методы распределенного рендеринга виртуальных сцен на GPU с реализацией эффекта размытия в движении

Предложены методы и подходы для распределенной визуализации трехмерных виртуальных сцен с имитацией эффекта размытия изображения, возникающего в результате высоких скоростей перемещения моделируемых средств наблюдения или объектов, находящихся в их поле зрения. Предлагаемые решения основаны на использовании параллельных вычислений на многоядерных GPU и обеспечивают рендеринг сложных сцен в реальном времени.

Ключевые слова: рендеринг, реальное время, виртуальная камера, графический процессор, размытие в движении

Введение

Системы виртуального окружения и имитационно-тренажерные комплексы получили широкое распространение в современном мире. Одной из базовых составляющих таких комплексов и систем является аппаратно-программный компонент для визуализации трехмерной виртуальной среды. Как правило, такая визуализация должна выполняться в масштабе реального времени, т. е. на все расчеты и формирование изображения отводится не более 40 мс. Это обусловлено необходимостью интерактивного взаимодействия оператора со средой и для реализации плавного движения динамических объектов в кадре. Для поддержания режима реального времени при рендеринге сложных виртуальных сцен требуются методы и алгоритмы, основанные на принципах распределенных вычислений и использующие современную программно-аппаратную базу, поэтому их разработка является весьма актуальной научной задачей.

В рамках повышения реалистичности результатов визуализации виртуальной среды, используемой в тренажерных системах и системах виртуального окружения, решается много задач, в том числе связанных с моделированием особенностей формирования изображения человеческим глазом и техническими устройствами наблюдения [1–3]. Одной из них является имитация на синтезируемом изображении эффектов, вызываемых высокими скоростями перемещения виртуального наблюдателя, моделируемого средства наблюдения или видимых ими объектов. Так, при движении на больших скоростях в реальной жизни происходит сужение эффективного поля зрения наблюдателя и концентрация внимания в узком угле вдоль направления взгляда. Размытие же объектов, попадающих в поле зрения реальной видеокамеры и движущихся с высокой скоростью, возникает

ввиду того, что время экспонирования каждого кадра обычно достаточно велико. Например, для съемки со стандартной частотой 25 кадров в секунду это время составляет 1/50 с. Имея большую скорость, объект за время формирования кадра проходит заметное расстояние и в результате выглядит в кадре смазанным.

Можно выделить два основных подхода для моделирования эффекта размытия в движении (*motion blur*). Один из них заключается в том, что смягчение изображения происходит только при быстром перемещении или повороте виртуальной камеры, остальные же объекты предполагаются статическими или имеющими незначительные скорости. Например, в работе [4] описаны методы шейдерной реализации эффекта размытия в движении путем постобработки изображения сцены с учетом буфера глубины и вычисления векторов скоростей изменения экранных координат точек объектов. Размытие проводится путем вычисления цвета каждого пикселя в виде усредненной суммы цветов нескольких выборок из исходного изображения, выполняемых вдоль вектора скорости с началом в данном пикселе. Преимуществами такого решения являются использование параллельных вычислений на графическом процессоре (GPU) и отсутствие изменений в основных шейдерах визуализации ввиду независимого выполнения стадии реализации эффекта. Однако в исходном виде возможная область его применения ограничена сценами со статическими объектами.

Другой подход предполагает реализацию эффекта размытия в движении с учетом скоростей динамических объектов любого типа, а не только камеры. В таком случае может наблюдаться как размытие изображения в целом, так и отдельных его регионов, содержащих быстродвижущиеся элементы сцены. Так, авторы работы [5] предлагают методы моделирования эффекта независимо для каждого динамического объекта на базе некоторого объема виртуаль-

ной среды ("объема движения"), занимаемого этим объектом в течение имитируемого времени экспонирования. Объем движения формируется путем аппроксимации, предполагающей деление объекта на заднюю и переднюю части (первая берется из положения объекта в момент начала экспонирования, вторая — в момент окончания) на основе вектора движения с дополнением центральной области путем анализа контуров. Положительные результаты такого решения состоят в отсутствии зависимости плавности размытия от скорости движения, а также в возможности применения эффекта только к требуемому набору объектов. Однако размытие кадра в случае движения и поворотов камеры авторами не предусматривается. Кроме того, описанные методы и алгоритмы используют только ограниченный функционал средств графической карты (буфер глубины) без участия параллельных вычислений в явном виде. Другие методы в рамках рассматриваемого подхода описаны в работе [6], ее авторы реализуют размытие в движении с помощью постобработки на GPU. Моделирование эффекта выполняется за два прохода рендеринга. Первый проход заключается в формировании списков экранных координат фрагментов, принадлежащих объектам сцены, между их положениями в соседних кадрах на основе алгоритма Брезенхэма. Во втором проходе оценивается видимость этих фрагментов в положениях, сохраненных в их списках. Все видимые фрагменты, соответствующие одному пикселу кадра, смешиваются путем вычисления усредненной суммы их цветов, которая формирует итоговый цвет рассматриваемого пикселя. Данные решения отличаются высоким качеством синтезируемого изображения и обеспечивают размытие при движении как объектов, так и виртуальной камеры, но имеют довольно значительную зависимость производительности от площади, занимаемой быстродвижущимся объектом в кадре.

В настоящей работе рассматриваются оригинальные методы и алгоритмы распределенной визуализации на GPU виртуальных объектов в масштабе реального времени с учетом размытия в движении, являющиеся развитием описанной в работе [4] технологии постобработки. Новизной предлагаемых решений являются возможность применения этой технологии в случае движений не только виртуальной камеры, но и объектов в ее поле зрения; использование многоцелевого шейдерного рендеринга (*multiple render targets*) для синтеза самого изображения кадра и записи необходимых дополнительных данных в виде текстур за один проход; авторский подход к расчету весовых коэффициентов для произвольного числа выборок вдоль векторов изменения экранных координат точек объектов.

1. Размытие изображения при движении камеры

Вначале рассмотрим реализацию смягчения изображения в рамках статических сцен, в которых изменение положения и ориентации доступно только виртуальной камере. Решение задачи основано на

распределенной постобработке на GPU изображений, получаемых в результате рендеринга трехмерной виртуальной сцены, с учетом информации из буфера глубины видеокарты. Содержимое буферов кадра и глубины при этом должны быть представлены в виде текстур T_{color} и T_{depth} , имеющих аналогичные буферам размеры $W \times H$ (в пикселях). Чтобы избежать лишних временных затрат на копирование информации, в данной работе применяется механизм рендеринга исходного кадра непосредственно в упомянутые текстуры с помощью технологии внеэкраных буферов FBO (*frame buffer object* [7]).

Для обработки кадра и отображения модифицированного изображения используется собственный фрагментный шейдер. Перед его применением необходимо установить ортографическую проекцию, а также выключить тест глубины и смешивание. Далее выполняется визуализация прямоугольника, размер которого совпадает с размером кадра, с наложением на него текстуры T_{color} . При расчете цвета каждого пикселя (i, j) данного прямоугольника, где $i \in [0, W]$, $j \in [0, H]$ (начало отсчета располагается в левом нижнем углу кадра), вычисляется двумерный вектор V изменения экранных координат точки P некоторого виртуального объекта, цвет C_0 и глубина $d \in [0, 1]$ которой записаны в пикселях (i, j) текстур T_{color} и T_{depth} . Для этого вначале необходимо восстановить координаты P_{wes} точки P в мировой системе координат (CK) WCS на основе ее экранных координат $P_{scr}(x, y, z, 1) = (i/W, j/H, d, 1)$, где $x, y, z \in [0, 1]$. При выполнении обратной операции, т. е. вычислении P_{scr} к P_{wes} применяются текущие видовая M_v (переводит из CK WCS в CK VCS виртуальной камеры) и проекционная M_{pr} (переводит в CK CCS отсечения) матрицы, перспективное деление на w -координату (переход к CK NDCS нормализованного объема видимости), а также преобразование координат из диапазона $[-1, 1]$ к диапазону $[0, 1]$. Математически это можно выразить следующим образом:

$$\begin{aligned} P_{ccs} &= M_{pr} M_v P_{wes} = \\ &= (wP_{ndcs,x}, wP_{ndcs,y}, wP_{ndcs,z}, w), \\ P_{ndcs} &= \frac{P_{ccs}}{P_{ccs,w}} = \frac{1}{w} P_{ccs}, \\ P_{scr} &= 0,5P_{ndcs} + (0,5, 0,5, 0,5). \end{aligned} \quad (1)$$

Отсюда искомые координаты P_{wes} найдем по формулам

$$\begin{aligned} P_{ndcs} &= 2P_{scr} - (1, 1, 1), \quad P_{ccs} = wP_{ndcs}; \\ P' &= w(M_{pr} M_v)^{-1} P_{ndcs} = \\ &= (ww'P_{wes,x}, ww'P_{wes,y}, ww'P_{wes,z}, ww'), \quad P_{wes} = \frac{P'}{P'_w} \end{aligned}$$

Рассчитав координаты P_{wes} точки P в CK WCS, вычислим ее экранные координаты P_{scr}^0 при предыдущем положении и ориентации виртуальной камеры. Для этого используем формулы (1), подставляя в них видовую матрицу M_v^0 , которая использовалась при синтезе прошлого кадра. Тогда направление искомого

вектора V изменения экранных координат точки P будет определяться разностью P_{scr} и P_{scr}^0 . С учетом параметра частоты f генерации кадров, 1/с, и значения t_{exp} времени экспонирования (выдержки, с), устанавливаемого для виртуальной камеры, будем вычислять V по формуле

$$V = \left(P_{scr} - P_{scr}^0 \right) f t_{exp}.$$

Для реализации размытия в движении определим во фрагментном шейдере цвет рассматриваемого пикселя (i, j) визуализируемого прямоугольника как взвешенную сумму $N \geq 4$ выборок из текстуры T_{color} проведенных вдоль вектора V с началом в пикселе (i, j) :

$$C = \sum_{k=0}^{N-1} w_k T_{color} \left(\left[i - k \frac{WV_x}{N} \right], \left[j - k \frac{HV_y}{N} \right] \right), \quad (2)$$

где $T_{color}(x, y)$ — цвет пикселя (x, y) текстуры T_{color} ; W и H — размеры текстуры, квадратные скобки обозначают взятие целой части числа. Расчет весовых коэффициентов w_k выполняется с учетом условий $\sum_{k=0}^{N-1} w_k = 1$ и $w_{N-1} = w_{min}$, где w_{min} — регулируемый вес самой удаленной от пикселя (i, j) выборки (в нашем случае используется $w_{min} = 0,1$). При этом формулы имеют вид

$$\begin{aligned} w_k &= w_{min} + (N - k - 1) \Delta w, \\ \Delta w &= \frac{2(1 - Nw_{min})}{N(N - 1)}. \end{aligned} \quad (3)$$

На рис. 1 и 2 даны примеры применения предложенного метода размытия при быстром перемещении и повороте виртуальной камеры.

2. Размытие быстродвижущихся объектов

В случае динамических сцен с движущимися объектами применение технологии постобработки кадров, описанной в разд. 1, также целесообразно, однако требуется некоторая модификация основного шейдера визуализации и шейдера эффекта. Это связано с изменением положений и ориентаций подвижных объектов с мировой CK WCS от кадра к кадру, что обуславливает необходимость включения модельных матриц M_m в расчет векторов V изменения экранных координат точек объектов. Если бы вычисление V по-прежнему проводилось в шейдере постобработки, то потребовалось бы идентифицировать, к какому именно объекту сцены принадлежит рассматриваемая точка P , а также передавать в шейдер большое число матриц M_m ввиду их уникальности для каждого объекта. Такое решение

было бы крайне неэффективным как по производительности, так и по использованию памяти.

В настоящей работе предлагается осуществлять распределенный расчет векторов V непосредственно в шейдерах, отвечающих за синтез исходного изображения виртуальной среды, видимого камерой и сохраняемого в текстуру T_{color} . Такие шейдеры в момент использования уже имеют информацию о текущем объекте рендеринга и всех необходимых матрицах, в том числе модельной.

Суть предлагаемого подхода, основанного на многоцелевом рендеринге, заключается в следующем. Перед визуализацией виртуальной сцены вместо стандартного экранного буфера вывода подключается специальный FBO-буфер, с которым ассоциированы две цели рендеринга в виде перезаписываемых текстур T_{color} и T_v . Размеры $W \times H$ этих текстур должны совпадать с текущими размерами кадра, а внутренний формат задается как GL_RGB16F , т. е. каждый канал представлен 16-битным вещественным числом. T_{color} как и ранее хранит RGB-изображение сцены, а в каждый пиксель (i, j) , $i \in [0, W]$, $j \in [0, H]$ текстуры T_v будет записываться вектор V для соответствующего пикселя (i, j) текстуры T_{color} . Данные буфера глубины на этапе постобработки не потребуются, поэтому текстура T_{depth} не используется.

При рендеринге каждого виртуального объекта активируется требуемый вершинный шейдер, кото-



Рис. 1. Изображение сцены при быстром движении камеры вперед в обычном режиме (а) и с эффектом размытия в движении (б)



Рис. 2. Изображение сцены при быстром повороте камеры вокруг ее вертикальной оси в обычном режиме (а) и с эффектом размытия в движении (б)

рый получает на вход координаты V_{ocs} вершин полигонов этого объекта, представленные в локальной СК OCS, текущую модельно-видовую $M_{mv} = M_v M_m$ (переводит координаты из СК OCS в видовую СК VCS камеры) и проекционную M_{pr} матрицы, а также модельно-видовую матрицу M_{mv}^0 , применявшуюся при синтезе прошлого кадра. Заметим, что M_{mv}^0 необходимо хранить для каждого объекта и перезаписывать текущей матрицей M_{mv} после его визуализации. В процессе выполнения вершинный шейдер вычисляет координаты V_{ccs} и V_{ccs}^0 поступающих вершин в СК CCS отсечения для текущего и предыдущего кадров по формулам

$$V_{ccs} = M_{pr} M_{mv} V_{ocs}, \quad V_{ccs}^0 = M_{pr} M_{mv}^0 V_{ocs},$$

после чего передает их на выход. Обработка вершин проводится параллельно на GPU. Далее графический конвейер выполняет для каждого треугольного полигона объекта интерполяцию этих координат с получением положений P_{ccs} и P_{ccs}^0 (в текущем и предыдущем кадрах соответственно) некоторой точки P объекта в СК CCS, которые поступают на вход фрагментного шейдера. В нем осуществляется преобразование этих положений в СК NDCS нор-

мализованного объема видимости путем перспективного деления на w -координату, а также расчет вектора V изменения координат точки P в экранном пространстве:

$$P_{ndcs} = \frac{P_{ccs}}{P_{ccs,w}}, \quad P_{ndcs}^0 = \frac{P_{ccs}^0}{P_{ccs,w}^0}, \quad V = \frac{(P_{ndcs} - P_{ndcs}^0) f t_{exp}}{2},$$

где f — частота генерации кадров, $1/c$; t_{exp} — значение выдержки, с, для камеры. Деление на 2 применяется для перехода из СК NDCS в экранное пространство, при котором оси координат масштабируются ровно в 2 раза. Вектор V вместе с рассчитываемым цветом фрагмента, соответствующего точке P текущего кадра, поступают на выход фрагментного шейдера и записываются в текстуры T_v и T_{color} .

После окончания рендеринга виртуальной сцены выполняется алгоритм постобработки изображения кадра, аналогичный тому, что описан в разд. 1, за исключением расчета вектора V . Для этого с помощью специального шейдера, который имеет доступ к текстурам T_{color} и T_v , визуализируется прямоугольник размером с кадр. Цвет каждого пикселя (i, j) этого прямоугольника вычисляется по формулам (2) и (3), где значения координат V_x и V_y вектора V считаются из пикселя (i, j) текстуры T_v . Пример реализации описанного подхода для размытия движущегося виртуального объекта представлен на рис. 3.

3. Результаты

На базе описанных решений были разработаны программные модули для систем визуализации трехмерных виртуальных сцен. Они обеспечивают имитацию эффекта размытия в движении на синтезируемых изображениях трехмерной среды, видимых виртуальной камерой. При реализации программных компонентов использовались возможности распределенных вычислений на современных многоядерных GPU и шейдерного языка GLSL последнего поколения. Полученные модули прошли успешную апробацию в составе системы визуализации GLView [8], разработанной в ФГУ ФНЦ НИИСИ РАН.

Оценка производительности разработанных решений проводилась путем измерения частоты генерации кадров (FPS) при визуализации виртуальной сцены трехмерного полигона с моделью квадрокоптера (рис. 4). Общее число треугольных полигонов данной сцены составляет около 600 тыс. Эксперименты проводили на персональном компьютере с процессором Intel Core i7 (3,5 ГГц) и видеoadаптером NVIDIA

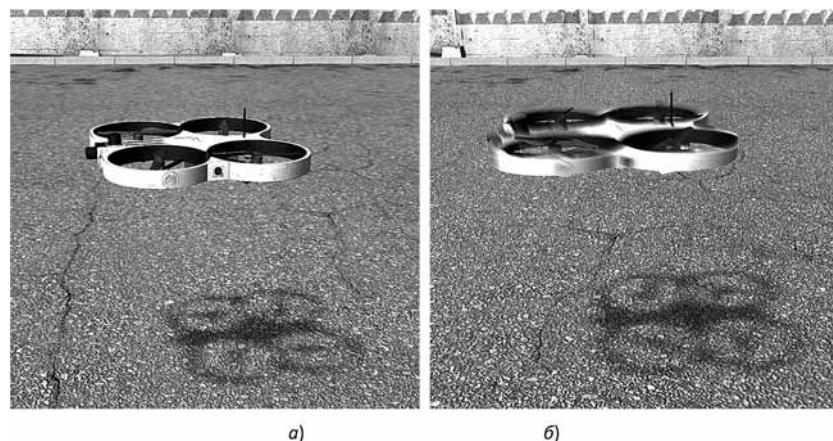


Рис. 3. Изображение быстродвижущегося объекта с эффектом размытия в движении (а) и без него (б)



Рис. 4. Виртуальная модель полигона с квадрокоптером

**Замеры FPS, кадров/с,
при визуализации сцены виртуального полигона**

Число N выборок из текстуры T_{color} для каждого вектора V	Разрешение изображения, пиксели		
	1280×720 (HD)	1920×1080 (Full HD)	3840×2160 (4K)
Не используется (без эффекта размытия)	212	175	129
4	188	171	127
8	187	168	126
16	185	166	124
32	180	159	121

GeForce GTX 970 (GPU содержит 1664 ядра). В ходе тестовых испытаний изменяли число N выборок (4, 8, 16 и 32) из текстуры T_{color} , проводимых вдоль вектора V изменения экранных координат точки объекта, а также итоговое разрешение получаемого изображения (форматы HD 1280×720, Full HD 1920×1080 и UHD 4K 3840×2160). Результаты проведенных экспериментов сведены в таблицу. Они показывают, что рендеринг высокополигональной сцены с моделированием эффекта размытия в движении на основе разработанных методов и алгоритмов выполняется с высокими значениями FPS, обеспечивающими масштаб реального времени даже при синтезе изображений в формате 4K. Снижение FPS по сравнению с визуализацией той же сцены без реализации эффекта размытия находится в допустимых пределах. Поэтому предложенные решения могут использоваться при создании систем визуализации для имитационно-тренажерных комплексов и систем виртуального окружения.

Заключение

Рассмотрено решение задачи моделирования эффекта размытия в движении на изображениях трехмерной виртуальной среды. Предложены методы и алгоритмы для случаев наблюдения виртуальной камерой за статическими и динамическими объектами. Разработанные решения основаны на широком использовании распределенных вычислений на современных многоядерных графических процессорах, что обеспечивает рендеринг трехмерных виртуальных сцен с имитацией эффекта размытия в движении в масштабе реального времени. Апробация предложенных методов и подходов продемонстрировала возможность их применения при реализации программного обеспечения имитационно-тренажерных комплексов и систем виртуального окружения.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 19-07-00393.

Список литературы

1. Kučík M., Zemčík P. Simulation of Camera Features // In Proceedings of the 16th Central European Seminar on Computer Graphics. — 2012. — P. 117–123.
2. Barsky B. A., Kosloff T. J. Algorithms for rendering depth of field effects in computer graphics // 12th WSEAS International Conference on COMPUTERS. — 2008. — P. 999–1010.
3. Мальцев А. В. Методы моделирования световых эффектов и искажений видеосигнала в виртуальных средствах наблюдения // Информационные технологии и вычислительные системы. — 2019. — Т. 9, № 2. — С. 54–61.
4. Rosado G. Motion Blur as a Post-Processing Effect // In GPU Gems 3. — 2007. — P. 575–582.
5. Wloka M. M., Zelezník R. C. Interactive real-time motion blur // The Visual Computer. — 1996. — Vol. 12, No. 6. — P. 283–295.
6. Choi J., Oh K. Real-time motion blur based on per pixel fragment list // Proceedings of the International Conference on Research in Adaptive and Convergent Systems. — 2017. — P. 132–135.
7. Framebuffer Object. URL: https://www.khronos.org/opengl/wiki/Framebuffer_Object (дата обращения 15.05.2020).
8. Михайлюк М. В., Торгашев М. А. Система визуализации "GLView" для имитационно-тренажерных комплексов и систем виртуального окружения // Труды 25-й Международной научной конференции Графикон. — 2015. — С. 96–101.

Methods for Distributed Rendering of Virtual Scenes on GPU with Motion Blur Implementation

A. V. Maltsev, avmaltcev@mail.ru, Scientific Research Institute for System Analysis of the Russian Academy of Sciences, Moscow, 117218, Russian Federation

Corresponding author:

Maltsev Andrey V., Ph.D., Leading Researcher, Scientific Research Institute for System Analysis of the Russian Academy of Sciences, Moscow, 117218, Russian Federation
E-mail: avmaltcev@mail.ru

*Received May 21, 2020
Accepted May 29, 2020*

The paper presents methods and approaches for distributed visualization of three-dimensional virtual scenes with imitation of motion blur effect arising because simulated surveillance devices or objects in their field of view move with

high speed. The effect is implemented by means of post-processing images of virtual environment on a GPU using a special fragment shader. This shader blurs an input image based on velocity vector (in screen space coordinate system) at each pixel. For static scenes, where only a virtual camera can move and rotate, the proposed solution is a completely separate stage executed after common scene rendering. Velocity vectors are computed from depth buffer data at this stage. In the case of dynamic virtual scenes, it also requires minor changes of the main fragment program to calculate velocity vectors and save them in a data texture read by motion blur shader. Based on methods and algorithms proposed in this work, software modules were implemented. Approbation of these modules in the visualization system, which was developed in Scientific Research Institute for System Analysis of the Russian Academy of Sciences, showed that created methods and algorithms for motion blur simulation meet high requirements of real-time visualization. The modules use distributed computing by means of multicore GPU, provide real-time rendering of complex virtual scenes, and can have practical application in visualization systems for simulation-training complexes, virtual environment systems etc.

Keywords: rendering, real time, virtual camera, graphics processor, motion blur

Acknowledgements: This research was carried out with the financial support of the Russian Foundation for Basic Research in the framework of the scientific project No. 19-07-00393.

For citation:

Maltsev A. V. Methods for Distributed Rendering of Virtual Scenes on GPU with Motion Blur Implementation, *Programmnaya Ingeneria*, 2020, vol. 11, no. 4, pp. 213–218

DOI: 10.17587/prin.11.213-218

References

1. Kučíš M., Zemčík P. Simulation of Camera Features, *In Proceedings of the 16th Central European Seminar on Computer Graphics*, 2012, pp. 117–123.
2. Barsky B. A., Kosloff T. J. Algorithms for rendering depth of field effects in computer graphics, *12th WSEAS International Conference on COMPUTERS*, 2008, pp. 999–1010.
3. Maltsev A. V. Methods for simulation of light effects and video signal distortions in virtual surveillance devices, *Informatsionnye tekhnologii i vychislitel'nye sistemy*, 2019, vol. 9, no. 2, pp. 54–61 (in Russian).
4. Rosado G. Motion Blur as a Post-Processing Effect, *In GPU Gems 3*, 2007, pp. 575–582.
5. Wloka M. M., Zeleznik R. C. Interactive real-time motion blur, *The Visual Computer*, 1996, vol. 12, no. 6, pp. 283–295.
6. Choi J., Oh K. Real-time motion blur based on per pixel fragment list, *Proceedings of the International Conference on Research in Adaptive and Convergent Systems*, 2017, pp. 132–135.
7. Framebuffer Object, available at: https://www.khronos.org/opengl/wiki/Framebuffer_Object (date of access 15.05.2020).
8. Mihayluk M. V., Torgashev M. A. The system of visualization "GLView" for simulators and virtual environment systems, *Trudy 25-i Mezhdunarodnoj nauchnoj konferencii Grafikon-2015*, 2015, pp. 96–101 (in Russian).

ИНФОРМАЦИЯ

Продолжается подписка на журнал "Программная инженерия" на второе полугодие 2020 г.

Оформить подписку можно в любом отделении Почты России, через подписные агентства или непосредственно в редакции журнала.

Подписной индекс по Объединенному каталогу

"Пресса России" — 22765

Сообщаем, что с 2020 г. возможна подписка на электронную версию нашего журнала через:

ООО "ИВИС": тел. (495) 777-65-57, 777-65-58; e-mail: sales@ivis.ru,

ООО "УП Урал-Пресс". Для оформления подписки (индекс 013312) следует обратиться в филиал по месту жительства — <http://ural-press.ru>

Адрес редакции: 107076, Москва, Стромынский пер., д. 4,
Издательство "Новые технологии",
редакция журнала "Программная инженерия"

Тел.: (499) 269-53-97. Факс: (499) 269-55-10. E-mail: prin@novtex.ru

Е. В. Орлова, д-р техн. наук, проф., ekorl@mail.ru, Уфимский государственный авиационный технический университет

Методы и модели анализа данных и машинного обучения в задаче управления производительностью труда

Рассмотрены вопросы повышения производительности труда на предприятии с учетом экономических, демографических, социальных факторов и субъективной информации о качестве здоровья его персонала. Предложена технология управления производительностью труда, основанная на поэтапной обработке количественных и качественных данных и моделировании на основе методов статистического анализа и машинного обучения. Разработанные на этом направлении технологии и модели обеспечивают поддержку принятия управленческих решений при планировании траекторий роста производительности труда.

Ключевые слова: анализ данных, машинное обучение, кластеризация, классификация, производительность труда

Введение

Повышение производительности труда, модернизация приоритетных несырьевых отраслей экономики в настоящее время являются не только необходимыми условиями для экономического роста, но и важными составляющими национальной безопасности России. Производительность труда является ключевым индикатором экономического развития страны и детерминантом уровня жизни населения. В рамках Национального проекта "Повышение производительности труда и поддержка занятости"¹ целевым индикатором является прирост производительности труда на предприятиях базовых несырьевых отраслей экономики. При этом решение проблемы повышения производительности труда, обеспечения конкурентоспособности выпускаемой продукции, а также стабильного роста и развития отдельных предприятий, отраслей и регионов в значительной степени зависит от сохранения здоровья и безопасности человека.

Как экономическая характеристика эффективности использования труда и результативности производственно-экономической деятельности предприятий и организаций, производительность труда показывает объем затрат труда, необходимый для производства единицы продукции. Традиционно факторы роста производительности труда объединяют в следующие группы: материально-технические, которые учитывают уровень инновационности техники и технологий, создание высокопроизводительных рабочих мест; организационно-экономические, определяющие уровень развития специализации и концентрации производства, применяющие тех-

ники бережливого производства, совершенствования структуры и объемов производства, повышения квалификации; социально-экономические, определяющие уровень оплаты труда и условия труда. Однако факторы, формирующие человеческий потенциал, такие как состояние и качество здоровья работников (как производительных сил производства), не рассматриваются в качестве целевых, воздействие на которые могут при взаимодействии с остальными важными компонентами и факторами качественно изменить в лучшую сторону производительность труда.

Необходимо отметить, что для тщательного анализа содержательных основ рассматриваемой проблемы на настоящее время отсутствуют методики количественной оценки влияния уровня и качества здоровья на производительность труда, не представлены рекомендации по управлению здоровьем в целях роста производительности труда и эффективности деятельности предприятий и организаций. При этом под здоровьем в данном исследовании понимается, следуя уставу и рекомендациям Всемирной организации здравоохранения [1], "состояние полного физического, душевного и социального благополучия, а не только отсутствие болезней и физических дефектов". Здоровье, наряду с профессиональными, социальными навыками, знаниями и возможностями, образует человеческий потенциал. Рост качества человеческого потенциала прямо влияет на показатели результативности деятельности предприятий, организаций, институтов и экономик в целом.

Состояние и качество здоровья человека как фактор производительности его труда в отечественных и зарубежных работах исследован недостаточно. Вероятно, эта проблема выйдет на передний план уже после того, как будут исчерпаны другие механизмы и источники роста эффективности труда: модернизация и цифровизация производства, совершенство-

¹ Паспорт национального проекта (программы) «Производительность труда и поддержка занятости» (утв. президентом Совета при Президенте РФ по стратегическому развитию и национальным проектам, протокол от 24.12.2018 № 16).

вание управления и пр. При этом важно отметить, что в условиях четвертой промышленной революции качество рабочей силы переходит из малозначимых в группу важнейших факторов управления производительностью труда.

Вопросы охраны здоровья персонала в контексте системы управления человеческими ресурсами на предприятиях представлены в работах [2–4]. В них проанализирована существующая система менеджмента качества охраны труда и предложены механизмы ее совершенствования. Здоровье как часть человеческого капитала, а здоровьесбережение как элемент системы корпоративной социальной ответственности бизнеса, оценка эффективности системы здравоохранения в целом на макроэкономическом уровне, анализ эффективности инвестиций в здоровьесбережение и их влияние на экономический рост представлены в работах [5, 6].

Качественный анализ на основе экспертных оценок влияния уровня здоровья персонала на производительность труда проводился в работах [7, 8], регулярно приводится в отчетах всемирной организации здравоохранения (ВОЗ) и организации экономического сотрудничества и развития (ОЭСР) [1, 9, 10]. Формированию подходов к росту производительности труда как к фактору конкурентоспособности предприятий и организаций, к выявлению лучших методик поиска, подготовки и продвижения талантливых сотрудников посвящены исследования [11–13], совокупность опирающихся показателей, характеризующих условия труда и ценностно-мотивационной среды, представлены в работе [14]. Выстраивание общей сбалансированной системы управления производственно-экономической деятельностью предприятия, включая мониторинг и управление эффективностью ресурсов, в том числе трудовых, дано в работах [15–17].

Резюмируя можно сказать, что проблема, связанная с изучением разнородных факторов производительности труда, в том числе и факторов здоровья персонала,

построением моделей, выявляющих тип и характер связей факторов, формированием однородных групп работников, обладающих похожими профилями производительности труда в целях дальнейшего управления ею, является в настоящее время актуальной и значимой.

1. Концептуальный подход к управлению производительностью труда

Предлагается подход, основанный на поэтапной обработке информации и моделировании, отражающем демографические, социальные факторы и факторы о качестве здоровья персонала, сформированный в целях управления производительностью труда. Концептуально схема технологии представлена на рис. 1.

Этап 1. Качественный анализ состояния здоровья работников. Осуществляется проведение сплошного обследования работников предприятия с использованием медицинских осмотров и анкетирования. Результатом этого этапа являются объективные данные о состоянии здоровья, а также субъективная информация о состоянии и качестве здоровья.

Этап 2. Выбор и обоснование факторов, влияющих на индивидуальную производительность труда работников. Проводится разведочный предварительный анализ данных, оценивается влияние факторов, отражающих социальные, демографические характеристики работников, а также факторов, определяющих влияние состояния и качества их здоровья на производительность труда на основе корреляционного и регрессионного анализа.

Этап 3. Формирование однородных групп работников и построение типичных профилей работников каждой группы. Формируются однородные группы работников с позиции схожести значений их демографических, социальных факторов и характеристик качества здоровья. Результатом данного этапа являются типичные профили работников, принадлежащих качественно однородной группе.



Рис. 1. Концептуальная схема технологии управления производительностью труда (ПТ)

Таблица 1

Исследуемые показатели, обозначения и диапазон значений

Характеристика	Обозначение	Значение переменной
Возраст	age	20...65
Пол	gender	Женский (0), мужской (1)
Семейное положение	mag	Не женат/не замужем (0), женат/замужем (1)
Образование	edu	Среднее специальное (0), высшее (1)
Число детей	child	0, 1, 2 и т. д.
Наличие хронических заболеваний	chron_dis	Нет (0), есть (1)
Частота плохого самочувствия	bad_healh	Не часто (0), часто (1)
Самооценка здоровья как слабое и неудовлетворительное	health_self-ass	Нет (0), да (1)
Присутствует ли неправильное питание	nutr	Нет (0), да (1)
Наличие вредных привычек	bad_hab	Нет (0), да (1)
Индивидуальная производительность труда	labor_perf	136...420 руб./ч

Этап 4. Разработка комплекса управленческих решений, направленных на улучшение состояния и качества здоровья персонала для каждой однородной группы.

Предложенная технология анализа и моделирования апробируется на крупном промышленном предприятии Республики Башкортостан. Анализ данных и моделирование осуществляются с использованием средств программного комплекса Statistica 10.0.

2. Разведочный анализ данных

Для анализа используется массив полученных на первом этапе технологии данных (табл. 1) о качественных и количественных показателях, отражающих социальные характеристики работников предприятия (образование, семейное положение, наличие и число детей), антропометрические характеристики (пол, возраст) и характеристики самооценки состояния и качества здоровья работников (наличие текущих проблем со здоровьем; наличие хронических заболеваний; самооценка здоровья; оценка правильности питания; наличие вредных привычек).

Перевод качественных значений признаков в количественные осуществляется на основе двоичного кодирования (0 и 1), при этом количественное значение признака увеличивается по мере интенсификации качественной его характеристики. Индивидуальная производительность труда рассчитывается по методике, которая используется для оценки производительности труда предприятий базовых несырьевых отраслей экономики¹. В рамках методики производительность труда отражает меру добавленной

стоимости, приходящуюся на одного сотрудника предприятия.

Для определения и визуального представления предполагаемого воздействия качественных признаков на производительность труда построены диаграммы размаха (рис. 2).

Диаграммы показывают, что разные уровни факторов gender, child, health_self-ass и bad_hab определяют разницу в показателе производительность труда — labor_perf. Так, мужчины имеют более высокую среднюю (около 280 руб./ч) производительность труда, чем женщины (225 руб./ч). У работников, у которых отсутствуют хронические заболевания, не часто наблюдается плохое самочувствие и отсутствуют вредные привычки, средний уровень производительности труда выше, чем у остальных. Работники с одним-двумя детьми имеют более стабильный уровень производительности труда, близкий к среднему, который, в свою очередь, выше, чем у работников без детей.

Диаграмма рассеяния в виде поверхности и карты линий уровня производительности труда labor_perf в зависимости от числовых переменных age и child (рис. 3, см. третью сторону обложки) позволяет визуализировать факт того, что более молодые работники в возрасте до 35 лет с детьми имеют наиболее высокие уровни производительности труда, чем возрастные работники (более 55 лет) с более чем тремя детьми, а также работники среднего возраста (от 35 до 55 лет) без детей.

Описательные статистики производительности труда (табл. 2) характеризуют наличие значительного размаха вариации и несовпадение обобщающих характеристик — средней арифметической, моды и медианы, а также коэффициентом вариации, отражающим пограничную однородность выборки данных.

Для выявления парных взаимосвязей факторов с производительностью труда проводится корреляционный анализ. Необходимо заметить, что обследуемые показатели измерены в разных шкалах — производительность труда, возраст и число

¹ Приказ Минэкономразвития России от 28.12.2018 № 748 (ред. от 15.10.2019) "Об утверждении Методики расчета показателей производительности труда предприятия, отрасли, субъекта Российской Федерации и Методики расчета отдельных показателей национального проекта "Производительность труда и поддержка занятости".

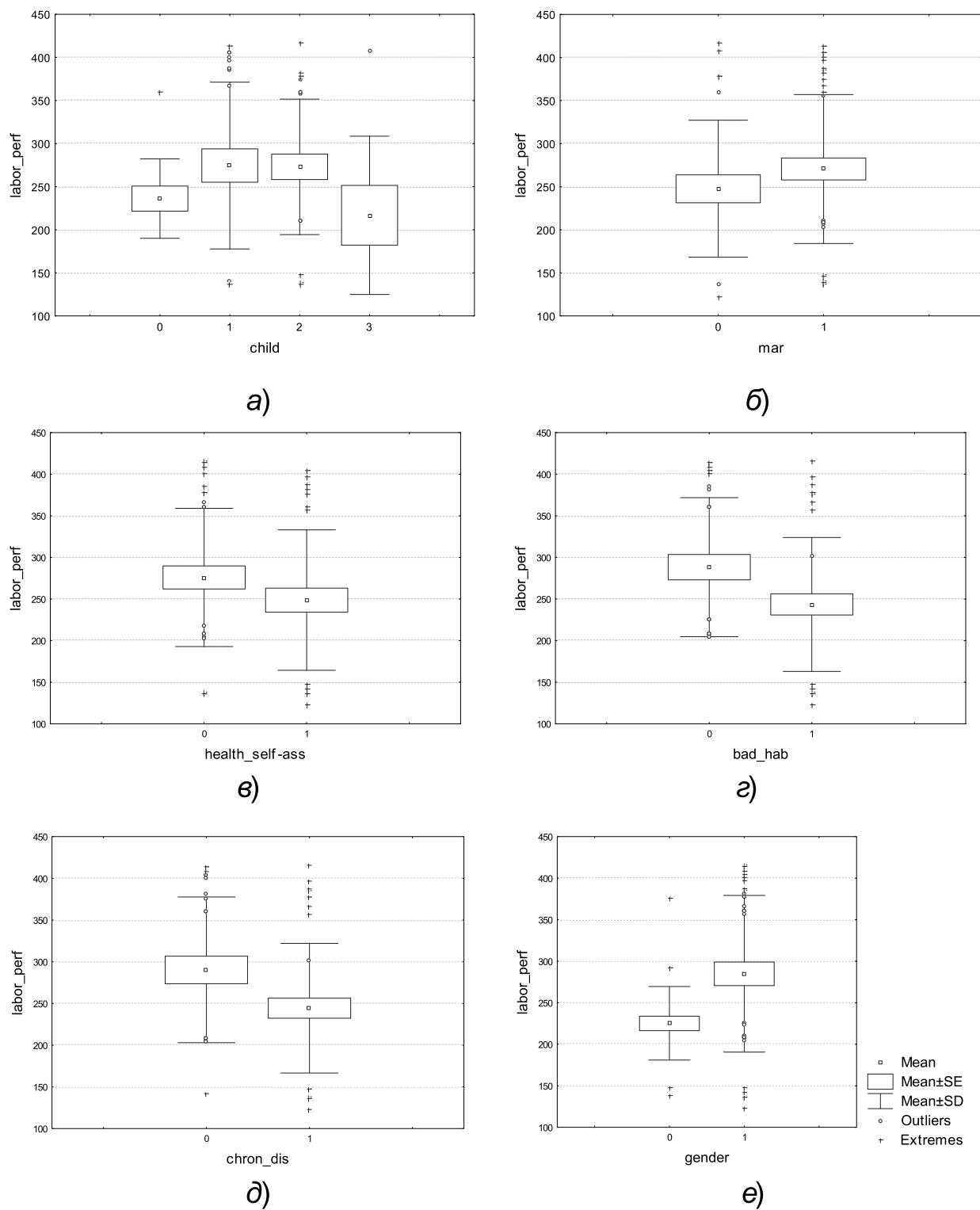


Рис. 2. Диаграммы размаха по группам факторов `child` и `labor_perf` (а), `mar` и `labor_perf` (б), `health_self-ass` и `labor_perf` (в), `bad_hab` и `labor_perf` (г), `chron_dis` и `labor_perf` (д), `gender` и `labor_perf` (е)

детей — в непрерывной метрической шкале, уровень образования — в порядковой (ранговой шкале), а остальные показатели — вnomинальной, дихотомической шкале. Поэтому анализ взаимосвязей факторов для выявления значимого влияния их на моделируемый показатель — производительность

труда — должен осуществляться с использованием разных метрик. Так, для измерения связи между производительностью труда, возрастом и числом детей используется коэффициент корреляции Пирсона, для оценки влияния уровня образования на производительность труда — коэффициент ранго-

Таблица 2

Описательные статистики показателя "производительность труда"

Показатель	Расчетное значение	Показатель	Расчетное значение
Среднее арифметическое	262,7	Минимальное значение	123
Мода	208	Максимальное значение	416
Медиана	231,5	Среднее квадратическое отклонение	84,2
		Коэффициент вариации	0,32

вой корреляции Спирмена, а для оценки влияния категориальных (дихотомических) переменных на производительность труда — многофакторный дисперсионный анализ.

С использованием многофакторного дисперсионного анализа (ANOVA) была проверена гипотеза о равенстве средних значений по соответствующим уровням пар факторов (производительность труда и наличие хронических заболеваний, производительность труда и пол и т. д. по всем семи дихотомическим переменным). Установлено, что гипотеза о равенстве средних значений была отклонена для трех факторов — пол (gender), наличие хронических заболеваний (chron_dis) и наличие вредных привычек (bad_hab). Для остальных пяти факторов такая гипотеза принимается, так как расчетные значения F-критерия Фишера меньше табличного при уровне значимости 0,05. Таким образом, факторы gender, chron_dis и bad_hab значимо связаны с производительностью труда labor_pert, а остальные четыре фактора не оказывают на нее статистически значимого влияния.

Оценка коэффициента Пирсона для пар возраст — производительность труда и число детей — производительность труда показала, что возраст оказывает статистически значимое обратное, среднее по своей силе, влияние на производительность труда (коэффициент корреляции Пирсона составляет $-0,32$), а число детей не оказывает сколько-нибудь значимого влияния на производительность труда (расчетное

значение t-критерия Стьюдента меньше табличного при уровне значимости 0,05). Расчет коэффициента корреляции Спирмена для оценки влияния уровня образования на производительность труда показал, что эти факторы не связаны корреляционной зависимостью, что противоречит общепринятому тезису о том, что качество и длительность образования работника способствуют росту его эффективности труда.

Для исключения ложных корреляций необходимо построить матрицу частных корреляций (табл. 3).

Из данных табл. 3 видно, что переменные age, gender, chron_dis, а также bad_hab значимо влияют на производительность труда (значимые зависимости отмечены в табл. 3 полужирным шрифтом). Кроме того, тесные частные корреляции наблюдаются между факторами chron_dis и bad_hab, bad_hab и health_self-ass, bad_hab и bad_health, nutr и health_self-ass, а также между парой факторов chron_dis и age, что объясняется тем, что наличие у работника плохого питания и вредных привычек вызывает хронические заболевания и плохое самочувствие.

Далее были протестированы несколько десятков регрессионных моделей разных спецификаций. При моделировании были учтены разные шкалы измерения исследуемых предикторов (факторов, предположительно влияющих на производительность труда), а также их множественные взаимодействия. Результаты моделирования по наилучшей модели даны в табл. 4. Модель регрессии включает совокуп-

Таблица 3

Матрица частных корреляций факторов (полужирным шрифтом отмечены значимые параметры)

Переменная	age	gender	mar	edu	child	chron_dis	bad_health	health_self-ass	nutr	bad_hab	labor_pert
age	1,00	-0,23	0,06	-0,22	0,30	0,35	-0,01	0,18	0,17	0,20	-0,32
gender	-0,23	1,00	0,20	0,07	0,03	0,04	0,18	-0,03	-0,10	0,01	0,34
mar	0,06	0,20	1,00	0,07	-0,00	-0,02	0,18	0,09	-0,04	0,07	0,03
edu	-0,22	0,07	0,07	1,00	-0,08	-0,28	-0,00	-0,05	0,00	-0,15	0,02
child	0,30	0,03	-0,00	-0,08	1,00	0,13	-0,14	0,20	0,13	0,26	-0,02
chron_dis	0,35	0,04	-0,02	-0,28	0,13	1,00	0,26	0,26	0,25	0,77	-0,27
bad_health	-0,01	0,18	0,18	-0,00	-0,14	0,26	1,00	0,48	0,21	0,31	0,02
health_self-ass	0,18	-0,03	0,09	-0,05	0,20	0,26	0,48	1,00	0,54	0,41	-0,31
nutr	0,17	-0,10	-0,04	0,00	0,13	0,25	0,21	0,54	1,00	0,18	-0,20
bad_hab	0,20	0,01	0,07	-0,15	0,26	0,77	0,31	0,41	0,18	1,00	-0,25
labor_pert	-0,32	0,34	0,03	0,02	-0,02	-0,27	0,02	-0,31	-0,20	-0,25	1,00

Таблица 4

Результаты регрессионного анализа производительности труда (полужирным шрифтом отмечены значимые параметры)

Предиктор регрессии	Коэффициент регрессии	Ошибки коэффициента регрессии	t-статистика	p-value
age	-5,232	2,2219	-2,35490	0,028854
child	56,130	27,1983	2,06372	0,052263
gender	70,954	68,5949	1,03438	0,313303
mar	30,099	66,2839	0,45409	0,654656
edu	42,684	72,9698	0,58496	0,565118
chron_dis	46,812	111,0978	0,42136	0,677989
bad_health	219,260	107,9100	2,03188	0,055664
health_ass	-268,820	130,3419	-2,06242	0,052398
nutr	-51,487	32,1415	-1,60188	0,124860
bad_hab	51,016	44,9827	1,13412	0,270156
gender*mar	29,871	36,2038	0,82508	0,419056
gender*edu	34,112	62,9763	0,54166	0,594034
mar*edu	-25,474	35,5743	-0,71609	0,482217
gender*chron_dis	23,970	59,2647	0,40445	0,690173
mar*chron_dis	-584,479	120,7443	-4,84064	0,000099
edu*chron_dis	-50,330	52,4964	-0,95872	0,349145
gender*bad_health	254,329	76,3480	3,33118	0,003329
mar*bad_health	111,577	49,7048	2,24479	0,036250
edu*bad_health	121,695	55,9648	2,17450	0,041843
chron_dis*bad_health	560,109	103,1680	5,42909	0,000026
gender*health_ass	-229,364	78,0956	-2,93696	0,008152
mar*health_ass	-57,145	61,9458	-0,92250	0,367263
edu*health_ass	-108,392	43,4358	-2,49546	0,021441
chron_dis*health_ass	-488,851	105,3426	-4,64059	0,000158
bad_health*health_ass	-163,115	113,1772	-1,44124	0,164992
gender*nutr	-19,127	31,3961	-0,60921	0,549244
mar*nutr	-9,016	41,7107	-0,21616	0,831054
edu*nutr	25,113	27,2835	0,92043	0,368315
chron_dis*nutr	52,730	63,6249	0,82876	0,417023
bad_health*nutr	-13,660	29,9965	-0,45538	0,653739
health_ass*nutr	-113,542	59,3250	-1,91389	0,070054
gender*bad_hab	-93,376	59,2220	-1,57671	0,130550
mar*bad_hab	516,338	108,2071	4,77176	0,000116
edu*bad_hab	83,426	66,6687	1,25135	0,225245
chron_dis*bad_hab	578,543	141,1459	4,09890	0,000558
bad_health*bad_hab	-476,916	95,9670	-4,96958	0,000074
health_ass*bad_hab	346,146	66,1466	5,23302	0,000040
nutr*bad_hab	48,252	42,7011	1,13000	0,271847
gender*mar*edu	38,035	66,6750	0,57045	0,574728
gender*mar*chron_dis	-11,878	75,8871	-0,15652	0,877190
gender*edu*chron_dis	15,825	26,1220	0,60580	0,551461
mar*edu*chron_dis	94,601	70,3018	1,34564	0,193478
gender*mar*bad_health	109,581	72,6311	1,50873	0,147004
gender*edu*bad_health	79,240	34,3925	2,30400	0,032080
mar*edu*bad_health	-39,764	100,9515	-0,39389	0,697831
gender*chron_dis*bad_health	-48,492	79,6383	-0,60890	0,549445
mar*chron_dis*bad_health	-60,264	119,5432	-0,50412	0,619689
edu*chron_dis*bad_health	-60,326	60,5823	-0,99578	0,331255
gender*mar*health_ass	-162,445	62,3059	-2,60721	0,016862
gender*edu*health_ass	-84,053	34,6773	-2,42387	0,024960

ность предикторов, в том числе мультипликативно связанных предикторов, имеющих линейные связи с моделируемым показателем производительность труда.

Построенная модель дает представление о количественном влиянии выбранных предикторов на производительность труда. Особенно усиливают свое негативное влияние совместные воздействия факторов. Так, максимально снижает производительность труда (в среднем на 477 руб./ч) такое сочетание факторов, как частота плохого самочувствия и наличие вредных привычек. Наличие хронических заболеваний и самооценка своего здоровья как недостаточно крепкое снижает производительность труда в среднем на 449 руб./ч, а у женатых мужчин этот эффект усиливается еще на 162 руб./ч. Также снижение производительности труда в среднем на 584 руб./ч наблюдается у женатых мужчин с хроническими заболеваниями.

Необходимо отметить существенные недостатки построенной регрессии. Во-первых, некоторые параметры модели являются статистически не значимыми, во-вторых, множество качественных предикторов негативно оказывается на результатах интерпретации модели, а также на сложности самой интерпретации.

3. Кластеризация и классификация работников

Для того чтобы сгладить выявленные неоднородности данных, а также упорядочить сложные взаимодействия факторов друг с другом необходима процедура разбиения имеющейся совокупности на однородные группы. Это позволит более подробно исследовать данные и выявить закономерности в полученных однородных группах. Для решения этой задачи тестируют разные методы и модели и выбирают наиболее адекватные.

Формирование однородных групп работников на основе кластеризации

Кластеризация проводится в два этапа — качественный анализ с помощью иерархических методов и анализ с помощью метода k -средних [18, 19]. Анализ различных разбиений исходной выборки методом иерархической классификации показал, что можно образовать от трех до шести кластеров. Для более обоснованной группировки объектов необходимо использование методов кластеризации, использующих количественные критерии для оценки качества разбиения. К таким методам относят метод k -средних.

Строго говоря, алгоритм k -средних (k -means) применим для разбиения на кластеры только числовых данных [20]. При наличии в исходных данных категориальных (качественных) переменных применяют модификации этого алгоритма — алгоритм k -modes и k -prototypes [20, 21]. Они отличаются тем, что используют другие меры близости объектов — процент несогласия, смешанное расстояние Евклида — Хэмминга. При этом во всех случаях сначала осуществляется процедура кодировки, т. е. преобразования значений качественных признаков в количественные (см. табл. 1). В данной работе применяется смешанное расстояние Евклида — Хэмминга, а в качестве функционала, отражающего критерий оптимальности разбиения и выраженного уровня желательности различных альтернативных разбиений в процессе кластерного анализа, выступает центроидный метод.

В табл. 5 представлены результаты кластеризации, в которой содержится четыре кластера ($k = 4$).

Итак, на данном этапе получена информация о числе кластеров работников, их размере, доле работников каждой группы в общей численности, информация о среднем возрасте работников каждой группы, среднем числе детей и среднем значении индивидуальной производительности труда. Самый многочисленный кластер (23 человека) — второй, составляет около трети всех опрошенных, это преимущественно замужние женщины с высшим образованием, возраст выше среднего, имеющие двух и более детей. Работники этого кластера имеют существенные проблемы со здоровьем, у ряда есть хронические заболевания, имеются вредные привычки и при этом работники данного кластера обладают самой низкой производительностью труда по выборке опрошенных.

Первый кластер образуют преимущественно женатые мужчины с высшим образованием, имеющие одного ребенка, следящие за своим здоровьем, хорошо питающиеся, у них отсутствуют проблемы со здоровьем. Средний возраст работников этой группы 34,9 лет, а их средняя индивидуальная производительность довольно высокая — 302,79 руб./ч. Данную категорию можно обозначить "здоровые" и не требующие особого внимания к своему здоровью.

Таблица 5
Центры кластеров (метод k -средних)

Показатель	Среднее значение показателя в кластере			
	1	2	3	4
age	34,9	45,22	35,61	33,10
gender	1	0	1	0
mar	1	1	1	0
edu	1	1	0	1
child	1	2	2	0
chron_dis	0	1	1	0
bad_healh	0	1	1	1
health_self-ass	0	1	0	1
nutr	0	1	0	1
bad_hab	0	1	1	0
labor_pert	302,79	222,39	291,78	226,9
Численность кластера	19	23	18	10

Третий кластер работников, составляющий 26 % от числа всех опрошенных, образуют преимущественно мужчины со средним образованием, имеющие двоих детей, они оценивают свое здоровье не очень высоко, нерегулярно питаются и часто имеют наряду с хроническими еще и сезонные заболевания. Четвертый кластер работников — это незамужние женщины с высшим образованием, не имеющие детей, у которых отсутствуют вредные привычки, а состояние своего здоровья они характеризуют как вполне удовлетворительное.

Моделирование с использованием методов машинного обучения

Для выбора наиболее эффективных методов, позволяющих решать задачи кластеризации и классификации работников предприятия по качеству их здоровья в целях увеличения производительности труда, в контексте второго этапа технологии дополнительно решается задача по сопоставлению эффективности разных методов машинного обучения и выбору одного из методов с максимальным значением критерия качества [21, 22].

Для этого решаются следующие две подзадачи.

1. Кластеризация работников в целях формирования однородных групп, имеющих похожие профили производительности труда. Используются следующие методы кластеризации: метод кластеризации k -средних и нейросетевой метод на основе самоорганизующихся карт Кохонена.

2. Классификация работников предприятия. Целью этой подзадачи является идентификация профиля производительности нового работника, или определение группы (кластера), к которой он относится с большей вероятностью. На данном этапе применяют следующие методы: растущие деревья решений, метод опорных векторов.

Решение первой подзадачи. Так как в задаче кластеризации число кластеров заранее неизвестно, а выборка работников достаточно разнородная, то для получения более обоснованного разбиения на кластеры используем для сравнения другой метод машинного обучения — *Automated Neural Networks* (ANN) и проведем сравнительную оценку эффективности двух методов для получения кластеров клиентов — метода k -средних и ANN-метода.

Для получения числа групп (кластеров) будем использовать *Self-organizing map* (самоорганизующиеся карты Кохонена) — нейронную сеть с обучением без учителя. Разделим исходную совокупность на три подвыборки — первая — обучающая выборка в объеме 70 % от совокупного объема выборки, используется для обучения нейронной сети и подстройки ее весов. Вторая подвыборка — тестовая, ее объем составляет 15 %, используется для проверки правильности обучения и переобучения. Третья подвыборка — выборка для валидации, служит для оценки точности нейронной сети на "новых данных". Сначала топологическая размерность сети задается 5×5 нейронов. В результате моделирования показано, что можно четко выделить четыре класса клиентов, так как именно четыре нейрона описали большую часть исходных данных. Результаты кластеризации, полученные на основе метода ANN, даны в табл. 6.

Таблица 6
Центры кластеров (ANN-метод)

Показатель	Среднее значение показателя в кластере			
	1	2	3	4
age	34,2	34,1	41,0	41,7
gender	1	0	1	1
mar	1	1	1	0
edu	1	1	1	1
child	1	1	2	2
chron_dis	0	0	1	0
bad_healh	0	1	1	1
health_self-ass	0	1	1	0
nutr	0	1	1	0
bad_hab	0	1	1	1
labor_pert	294,12	287,35	224,46	265,26
Численность кластера	17	14	24	15

Качественная картина полученного разбиения похожа на результат кластерного анализа на основе k -средних. По-прежнему самый многочисленный кластер работников — это работники среднего возраста, имеющие проблемы со здоровьем и невысокую среднюю производительность труда.

Кластер работников с высокой производительностью образуют молодые женатые мужчины, не имеющие проблем со здоровьем. Два промежуточных кластера образуют работники, не имеющие хронических заболеваний, они имеют среднюю производительность труда на уровне 265,26...287,35 руб./ч. Обращает на себя внимание тот факт, что более высокий уровень производительности труда присущ работникам, не имеющим хронических заболеваний (кластеры 1, 2 и 4). При использовании ANN-метода в однородные группы попали работники с большим размахом вариации переменной "возраст", о чем свидетельствуют их полученные средние значения — в кластерах 1 и 2, 3 и 4 почти одинаковые средние значения. При этом корреляционный анализ выявил явную отрицательную зависимость возраста и производительности труда.

Для более обоснованного выбора одного из методов кластеризации и получения однородных групп работников был проведен анализ устойчивости используемых методов. Для этого в выборку данных была введена дополнительная количественная переменная, отражающая число дней, проведенных работником на больничном листе. Далее с учетом этого нового фактора сформированы новые разбиения на кластеры. Кластеризация на основе k -средних показала большую стабильность групп с позиций их одинакового состава до и после введения нового

фактора, чем кластеризации на основе ANN-метода. Поэтому для решения поставленной задачи кластеризации работников и формирования их однородных групп наиболее целесообразно использовать метод k -средних.

Получив четыре однородных класса работников, можно сформировать их профили (как совокупность значений характеристик, однозначно отличающих работников разных кластеров друг от друга) для дальнейшего обоснованного построения адекватных стратегий управления производительностью труда. Возникает задача определения группы и соответственно профиля, который будет иметь новый работник, изначально отсутствующий в исходной выборке данных. Для решения этой задачи применяется классификация с учетом уже обоснованных четырех кластеров работников. Для этого необходимо выбрать методы и определить их сравнительную эффективность при проведении классификации, под которой понимается отнесение каждого из заданного множества объектов (работников) к одному из заранее известных классов.

Построены две модели классификации. Первая модель использует алгоритм машинного обучения на основе растущих деревьев решений (*Boosted trees*), а вторая модель использует метод опорных векторов (*Support Vector Machines, SVM*). Оценка качества классификации представляется числом верных предсказаний — кластера, к которому относится работник в тестовой выборке. Так в модели на основе растущих деревьев решений получено 96,4 % верных отнесений работников к кластерам, а в методе на основе опорных векторов получено 100 % правильных предсказаний. Показано, что SVM-метод в условиях множества категориальных (качественных) предикторов — все переменные о состоянии здоровья, уровень образования — прогнозирует с высокой точностью класс работника, т. е. его профиль. Метод случайных и растущих деревьев наиболее пригоден для прогнозирования, когда имеется множество количественных предикторов.

Сформированные на данном этапе технологии однородные группы работников, имеющих содержательно разные профили, дают основание для разработки дифференцированного пакета управленческих решений (стратегии) по управлению производительностью труда путем сохранения здоровья работников. Такие стратегии разрабатываются отдельно для каждого из четырех однородных кластеров. Управленческие решения могут быть направлены на текущий контроль, мониторинг и профилактику здорового образа жизни.

Заключение

Работа, описанная в настоящей статье, направлена на обеспечение поддержки принятия управленческих решений при планировании механизмов и проектов, целью которых является рост производительности труда и эффективности производственно-экономической деятельности предприятия. Для формирования пакета компьютерных программ,

обеспечивающих решение этой задачи, необходимо на содержательном уровне определиться с составом используемых алгоритмов и методов. Поиск эффективных алгоритмов и методов и составляет содержание настоящей работы. Базируясь на полученных результатах, системные аналитики и топ-менеджеры получают возможность выбора соответствующего инструментария или разработки нового для интеграции его в информационную систему управления предприятием.

Показано, что для более полного и всестороннего описания производительности труда как объекта управления кроме экономических факторов необходимо учитывать социальные, демографические и факторы, характеризующие здоровье персонала. С использованием методов корреляционного и регрессионного анализа установлено, что существенными детерминантами производительности труда являются показатели состояния и качества здоровья работников и в случае отсутствия негативных проявлений способствуют, при прочих равных условиях, росту производительности труда.

Для того чтобы обоснованно формировать решения по управлению производительностью труда с учетом совокупности выявленных факторов необходим комплексный подход, который системно описывает закономерности их влияния на эффективность труда и позволяет формировать адекватные управленческие решения. Предложена технология управления производительностью труда на основе решений, связанных с улучшением состояния и качества здоровья персонала. В рамках технологии осуществляется выбор и обоснование факторов на основании их влияния на производительность труда, кластеризация работников и формирование их однородных групп с позиции схожих значений демографических, социальных факторов и характеристик качества здоровья с дальнейшим формированием типичных профилей работников в однородных группах. Это является основой для разработки дифференцированных стратегий управления, ориентированных на работников каждой группы.

Доказано, что для решения задачи кластеризации работников и формирования их однородных групп наиболее целесообразно использовать метод k -средних, имеющий большую устойчивость по сравнению с методом кластеризации на базе нейронных сетей Кохонена. В решаемой задаче, содержащей множество качественных переменных, таких как пол, образование, качественная самооценка здоровья для идентификации профиля новых работников предприятия большую эффективность продемонстрировал метод классификации на основе метода опорных векторов.

Список литературы

1. Improving Health Worker Productivity and Performance in the Context of Universal Health Coverage: The Roles Of Standards, Quality Improvement, And Regulation — Final Draft. Global Health Workforce Alliance (Ghwa). Technical Working Group (TWG) #7. 2014. URL: <https://www.who.int/workforcealliance/>

- media/news/2014/WG7_Productivityandperformance.pdf (дата обращения 15.02.2020).
2. Галкина Е. Е., Кабанов А. С., Ханецкий А. С. Некоторые аспекты оценки эффективности системы менеджмента охраны труда и техники безопасности на предприятиях авиакосмического комплекса // Качество и жизнь. — 2018. — № 1 (17). — С. 55–60.
 3. Сударкина Х. В., Глод О. Д. Трудоохранный менеджмент: система, методы и функции управления охраной труда на предприятии // Инновационная наука. — 2016. — № 6-1. — С. 245–248.
 4. Чижова М. А., Хайруллина Л. И. Системные действия в управлении охраной труда: менеджмент безопасности и охраны здоровья работников // Вестник технологического университета. — 2016. — Т. 19, № 24. — С. 163–167.
 5. Bloom D. E., Canning D., Sevilla J. The Effect of Health on Economic Growth: A Production Function Approach // World Development. — 2004. — Vol. 32, Issue 1. — P. 1–13.
 6. Sheiner L., Malinovskaya A. Measuring Productivity in Healthcare: An Analysis of The Literature. Hutchins Center on Fiscal and Monetary Policy at Brookings. 2016. URL: https://www.brookings.edu/wp-content/uploads/2016/08/hp-lit-review_final.pdf (дата обращения 15.02.2020).
 7. Огородова М. В. Повышение производительности труда на основе интеграции промышленного производства и здоровьесбережения: монография. — Нижний Новгород: Изд-во ВГИПА, 2005. — 146 с.
 8. Черепов В. М., Ефремов Д. В., Калинина М. Ю. Производительность труда и состояние здоровья работника // Проблемы социальной гигиены, здравоохранения и истории медицины. — 2017. — № 25(6). — С. 324–327.
 9. The Labour Market for Human Resources for Health in Low – and Middle-Income Countries. The World Health Organization (WHO). 2012. URL: http://www.who.int/hrh/resources/Observer11_WEB.pdf (дата обращения 15.02.2020).
 10. Job Quality, Health and Productivity. An evidence-based framework for analysis OECD Social, Employment and Migration Working Papers. 2018. No. 221. URL: <https://www.oecd-ilibrary.org/docserver/a8c84d91-en.pdf?expires=1590837940&id=id&accn=ame=guest&checksum=060F8C11565E10A8106658C46B3A857A> (дата обращения 15.02.2020).
 11. Майер Д., Лайкер Дж. Талантливые сотрудники. Воспитание и обучение людей в духе Дао Toyota. — М.: Альпина Паблишер, 2007. — 296 с.
 12. Syarnubi A., Sembiring R. A., Siswara T., Zuraida R. Performance, productivity, and safety & health among employee of oil & gas company in Qatar // IOP Conf. Series: Earth and Environmental Science. — 2018. — Vol. 195. — P. 012075.
 13. Бородулин А. Л. Взаимосвязь потока создания талантливых сотрудников и программ повышения производительности труда // Социально-трудовые исследования. — 2019. — № 4 (37). — С. 92–97.
 14. Сладкова Н. М., Ильченко О. И. Производительность труда: подход к разработке типовых опережающих показателей результаториентированной системы труда // Социально-трудовые исследования. — 2019. — № 3 (6). — С. 121–133.
 15. Orlova E. V. Synergic Approach for the Coordinated Control in Production and Economic System // Proceedings of the 30th International Business Information Management Association Conference (IBIMA). Vision 2020: Sustainable Economic development, Innovation Management, and Global Growth, Madrid, 2017. — P. 704–712.
 16. Orlova E. V. Technology for Control an Efficiency in Production and Economic System // Proceedings of the 30th International Business Information Management Association Conference (IBIMA). Vision 2020: Sustainable Economic development, Innovation Management, and Global Growth, Madrid, 2017. — P. 811–818.
 17. Orlova E. V. Modeling and Coordinated Control for the Production and Economic System // Proceedings of the Mathematical Modeling Session at the International Conference Information Technology and Nanotechnology (MM-ITNT 2017), CEUR Workshop Proceedings. 2017. — Vol. 1904. — P. — 1–6.
 18. Orlova E. V. Model for Operational Optimal Control of Financial Recourses Distribution in a Company // Computer Research and Modeling. — 2019. — Vol. 2. — P. 343–358.
 19. Huang Z. Extensions to the k-means algorithm for clustering large data sets with categorical values // Data mining and knowledge discovery. — 1998. — Vol. 2, No. 3. — P. 283–304.
 20. Jinchao Ji, Wei Pang, Chunguang Zhou, Xiao Han, Zhe Wang. A fuzzy k-prototype clustering algorithm for mixed numeric and categorical data // Knowledge-Based Systems. — 2012. — Vol. 30. — P. 129–135.
 21. Orlova E. V. Decision-Making Techniques for Credit Resource Management Using Machine Learning and Optimization // Information. — 2020. — Vol. 11 (3). — P. 1–17.
 22. Shalev-Shwartz Shai, Shai Ben-David. Understanding Machine Learning: From Theory to Algorithms. Cambridge: Cambridge University Press, 2014. — 499 p.

Methods and Models of Data Analysis and Machine Learning in the Problem of Labor Productivity Management

E. V. Orlova, ekorl@mail.ru, Ufa State Aviation Technical University, Ufa, 450008, Russian Federation

Corresponding author:

Orlova Ekaterina V., Dr. Sc., Professor, Ufa State Aviation Technical University, Ufa, 450008, Russian Federation,
E-mail: ekorl@mail.ru

Received on April 18, 2020
Accepted on May 22, 2020

The article considers the problem of labor productivity growth of the enterprise, taking into account economic, demographic, social factors and subjective information about the quality of staff health.

A critical analysis of existing approaches, methods and models in this area has been carried out and a number of significant shortcomings of the presented approaches has been identified that limit the scope of their application. There are no methods for quantifying the impact of health on labor productivity and developing further recommendations for managing employee's health in order to increase their labor productivity.

The technology for labor productivity management has been developed based on the phased processing and modeling of quantitative and qualitative data taking into account objective information about economic, demographic, social factors and subjective information about the factors of the health quality. The technology is based on statistical analysis and machine learning, and support managerial decision-making in planning health-saving strategies to increase labor productivity.

It is proved that to solve the problem of employee clustering and forming their homogeneous groups, it is most advisable to use the k-means method, which is more stable compared to the clustering method based on Kohonen neural networks. In the problem which contains many qualitative variables, such as gender, education, high-quality self-esteem of health for identifying the profile of new employees, the classification method based on the support vector method has demonstrated great efficiency.

Keywords: data analysis, machine learning, classification, clustering, labor productivity

For citation:

Orlova E. V. Methods and Models of Data Analysis and Machine Learning in the Problem of Labor Productivity Management, *Programmnaya Ingeneria*, 2020, vol. 11, no. 4, pp. 219–229

DOI: 10.17587/prin.11.219-229

References

1. **Improving** Health Worker Productivity And Performance In The Context Of Universal Health Coverage: The Roles Of Standards, Quality Improvement, And Regulation — Final Draft. Global Health Workforce Alliance (Ghwa). Technical Working Group (TWG) #7, 2014, available at: <https://www.who.int/> (date of access 15.02.2020).
2. **Galkina E. E., Kabanov A. S., Khanetsky A. S.** Some aspects of evaluating the effectiveness of the labor protection and safety management system at the enterprises of the aerospace complex, *Kachestvo i zhizn'*, 2018, no. 1 (17), pp. 55–60 (in Russian).
3. **Sudarkina H. V., Glod O. D.** Labor protection management: the system, methods and functions of labor protection management at the enterprise, *Innovatsionnaya nauka*, 2016, no. 6–1, pp. 245–248 (in Russian).
4. **Chizhova M. A., Khayrullina L. I.** Systemic actions in the management of labor protection: management of safety and health protection of workers, *Vestnik Tekhnologicheskogo universiteta*, 2016, vol. 19, no. 24, pp. 163–167 (in Russian).
5. **Bloom D. E., Canning D., Sevilla J.** The Effect of Health on Economic Growth: A Production Function Approach, *World Development*, 2004, vol. 32, issue 1, pp. 1–13.
6. **Sheiner L., Malinovskaya A.** *Measuring Productivity In Healthcare: An Analysis Of The Literature*. Hutchins Center on Fiscal and Monetary Policy at Brookings, 2016, available at: https://www.brookings.edu/wp-content/uploads/2016/08/hp-lit-review_final.pdf (date of access 15.02.2020).
7. **Ogorodova M. V.** *Increasing labor productivity through the integration of industrial production and health conservation: a monograph*, Nizhny Novgorod, VGIPA Publishing House, 2005. 146 p. (in Russian).
8. **Cherepov V. M., Efremov D. V., Kalinina M. Yu.** Labor productivity and employee health, *Problemy sotsial'noy gigiyeny, zdravookhraneniya i istorii meditsiny*, 2017, no. 25 (6), pp. 324–327 (in Russian).
9. **The Labour** Market for Human Resources for Health in Low – and Middle-Income Countries. The World Health Organization (WHO), 2012, available at: http://www.who.int/hrh/resources/Observer11_WEB.pdf (date of access 15.02.2020).
10. **Job Quality**, Health and Productivity. An evidence-based framework for analysis OECD Social, Employment and Migration. Working Papers, 2018, no. 221, available at: <https://www.oecd-ilibrary.org/> (date of access 15.02.2020).
11. **Meyer D., Liker J.** *Talented employees. Education and training of people in the spirit of Tao Toyota*, Moscow, Alpina Publisher, 2007, 296 p. (in Russian).
12. **Syarnubi A., Sembiring R. A., Siswawa T., Zuraida R.** Performance, productivity, and safety & health among employee of oil & gas company in Qatar, *IOP Conf. Series: Earth and Environmental Science*, 2018, vol. 195, pp. 012075.
13. **Borodulin A. L.** The relationship of the flow of creating talented employees and programs to increase labor productivity, *Sotsial'no-trudovyye issledovaniya*, 2019, no. 4 (37), pp. 92–97 (in Russian).
14. **Sladkova N. M., Ilchenko O. I.** Labor productivity: an approach to the development of typical operational indicators of a result-oriented labor system, *Sotsial'no-trudovyye issledovaniya*, 2019, no. 6 (3), pp. 121–133 (in Russian).
15. **Orlova E. V.** Synergetic Approach for the Coordinated Control in Production and Economic System, *Proceedings of the 30th International Business Information Management Association Conference (IBIMA). Vision 2020: Sustainable Economic development, Innovation Management, and Global Growth*, Madrid, 2017, pp. 704–712.
16. **Orlova E. V.** Technology for Control an Efficiency in Production and Economic System, *Proceedings of the 30th International Business Information Management Association Conference (IBIMA). Vision 2020: Sustainable Economic development, Innovation Management, and Global Growth*, Madrid, 2017, pp. 811–818.
17. **Orlova E. V.** Modeling and Coordinated Control for the Production and Economic System, *Proceedings of the Mathematical Modeling Session at the International Conference Information Technology and Nanotechnology (MM-ITNT 2017)*, CEUR Workshop Proceedings, 2017, vol. 1904, pp. 1–6.
18. **Orlova E. V.** Model for Operational Optimal Control of Financial Recourses Distribution in a Company, *Computer Research and Modeling*, 2019, vol. 2, pp. 343–358.
19. **Huang Z.** Extensions to the k-means algorithm for clustering large data sets with categorical values, *Data mining and knowledge discovery*, 1998, vol. 2(3), pp. 283–304.
20. **Jinchao Ji, Wei Pang, Chunguang Zhou, Xiao Han, Zhe Wang.** A fuzzy k-prototype clustering algorithm for mixed numeric and categorical data, *Knowledge-Based Systems*, 2012, vol. 30, pp. 129–135.
21. **Orlova E. V.** Decision-Making Techniques for Credit Resource Management Using Machine Learning and Optimization. *Information*, 2020, vol. 11(3), pp. 1–17.
22. **Shalev-Shwartz Shai, Shai Ben-David** *Understanding Machine Learning: From Theory to Algorithms*, Cambridge, Cambridge University Press, 2014, 499 p.

П. Н. Девягин, д-р техн. наук, проф., гл. науч. сотр., pdevyanin@astralinux.ru,
М. А. Леонова, науч. сотр., mleonova@astralinux.ru, ООО "РусБИТех-Астра", г. Москва

Применение подтипов и тотальных функций формального метода Event-B для описания и верификации МРОСЛ ДП-модели

Рассматриваются приемы описания мандатной существенно-ролевой ДП-модели управления доступом и информационными потоками в операционных системах семейства Linux (МРОСЛ ДП-модели) в математической нотации и формализованной нотации с использованием подтипов и тотальных функций на формализованном языке метода Event-B. Целью такого описания является упрощение дедуктивной верификации этой модели с применением инструментального средства Rodin и обеспечение более точного соответствия модели ее реализации непосредственно в программном коде операционной системы специального назначения Astra Linux Special Edition.

Ключевые слова: формальная модель управления доступом, верификация, Event-B, операционная система

Введение

Для проектирования, реализации и обоснования безопасности механизма управления доступом, входящего в состав многих средств защиты информации, таких как операционные системы (ОС), СУБД, используют формальные модели управления доступом. Одним из первых примеров этого является разработка в 1976 г. механизма мандатного управления (*Mandatory Access Control*) доступом ОС Multics на основе классической модели Белла — Лападулы [1, 2]. Другим заметным примером стала реализация на основе модели Биба [2, 3] механизма мандатного контроля целостности (*Mandatory Integrity Control*) в ОС семейства Microsoft Windows Vista/7/10 [4]. Можно также утверждать, что использованное в модели типизированной матрицы доступов [5] задание типов для объектов и субъектов доступа стало прообразом типизированного управления доступом (*Type Enforcement*), реализованного в распространенном пакете безопасности Security Enhanced Linux (SELinux) [6].

Из отечественной практики разработки средств защиты информации целесообразно отметить мандатную существенно-ролевую ДП-модель управления доступом и информационными потоками в ОС семейства Linux (сокращенно — МРОСЛ ДП-модель) [7], изначально созданную для реализации в защищенной операционной системе специального назначения (ОССН) Astra Linux Special Edition [8]. Эта модель обеспечивает сочетание мандатного и ролевого управления доступом с мандатным контролем целостности и включает описание условий безопасности, для которых в рамках модели математически доказано, что при их выполнении моделируемая система будет безопасной.

Этот факт означает, что невозможен несанкционированный захват недоверенными субъект-сессиями

(процессами ОССН) управления доверенными субъект-сессиями системы, а также невозможно возникновение запрещенных информационных потоков по памяти или по времени от сущностей (файлов, каталогов ОССН) с высоким уровнем конфиденциальности к сущностям с низким уровнем конфиденциальности.

Несмотря на достигнутые с момента начала в 2012 г. разработки МРОСЛ ДП-модели результаты по ее формированию и внедрению в ОССН, она продолжает развиваться и дорабатываться. Связано это со следующими причинами.

Во-первых, в модели должны учитываться изменения, вносимые в механизмы защиты ОССН. Например, начиная с версии 1.6, в ней реализован полнофункциональный мандатный контроль целостности с невырожденной решеткой уровней целостности [7], что было необходимо предварительно отразить и исследовать в рамках модели.

Во-вторых, превысивший сотни страниц объем описания модели, которое изначально осуществлялось на языке, принятом в математике (в математической нотации), потребовал поиска путей повышения эффективности разработки, проверки корректности и отсутствия ошибок в самой модели. Найденным здесь решением стал перевод описания модели на формализованный язык метода Event-B (в формализованную нотацию) и ее автоматизированная дедуктивная верификация с применением инструментального средства Rodin [9]. При этом элементам, задающим в рамках МРОСЛ ДП-модели состояния системы, в формализованной нотации ставятся в соответствие переменные Event-B, а правилам перехода из состояний в состояния — события Event-B. Инварианты на переменные описывают свойства внутренней согласованности элементов МРОСЛ ДП-модели. С помощью Rodin доказывается, что любой переход

из состояния в состояние, заданный событиями Event-B, сохраняет все инварианты состояния, что позволяет убедиться в корректности самой модели, а также доказательства выполнения условий безопасности системы.

В-третьих, формируемая в рамках МРОСЛ ДП-модели теория продолжает развиваться чтобы создать на перспективу условия для внедрения в ОССН новых механизмов защиты или расширения охватываемых ими компонентов ОССН. К примеру, в модель были добавлены запрещающие роли (т. е. роли, наличие прав доступа у которых не разрешает, а наоборот, запрещает предоставление субъект-сессиям соответствующих доступов) [10]. Так же в нее включены элементы, позволяющие моделировать управление доступом штатной СУБД PostgreSQL [11].

Перечисленные причины привели к необходимости постановки задачи по исследованию и поиску приемов согласованного описания МРОСЛ ДП-модели в математической и формализованной нотациях в целях учета постоянно вносимых в модель изменений, сокращению ресурсов, затрачиваемых на ее верификацию, и обеспечению более точного соответствия модели ее реализации непосредственно в программном коде ОССН.

Особую актуальность решению этой задачи придало то обстоятельство, что разработка формальной модели управления доступом и ее верификация с применением инструментальных средств стала частью требований, соответственно, уровней 4 и 3 доверия согласно вступивших в силу с 1 июня 2019 г. [12] утвержденных приказом Федеральной службы по техническому и экспортному контролю (ФСТЭК) России от 30 июля 2018 г. № 131 "Требований по безопасности информации, устанавливающих уровни доверия к средствам технической защиты информации и средствам обеспечения безопасности информационных технологий".

В настоящей работе проанализирован предлагаемый авторами ряд приемов, направленных на решение поставленной выше задачи исследования. Такие приемы основаны на использовании при описании и дедуктивной верификации МРОСЛ ДП-модели подтипов и тотальных функций формализованного языка метода Event-B. Эти приемы также могут быть полезны при разработке и верификации других формальных моделей управления доступом.

Анализ приемов, используемых для иерархического представления МРОСЛ ДП-модели

Основой для поиска путей согласованного описания МРОСЛ ДП-модели в математической и формализованной нотациях, а также "приближения" этого описания к реализации модели в программном коде ОССН, стал ее переход из монолитного в иерархическое представление [7, 13]. Это объясняется тем, что использованное на начальном этапе разработки модели наследованное от классических моделей монолитное представление предполагает описание ее элементов, правил перехода из состояний в состояния моделируемой системы, условий безопасности без разделения по видам управления доступом, к ко-

торым они непосредственно относятся. В результате, затрудняются как описание и верификация модели, так и обеспечение ее соответствия механизму защиты ОССН, в котором каждый вид управления доступом — дискреционный (ролевой) и мандатный (управления доступом и контроля целостности) в программном коде реализуется отдельными функциями. В иерархическом представлении модель описывается "по слоям" — уровням, при этом каждый нижний уровень представляет абстрактную систему, элементы которой не зависят от новых элементов, принадлежащих более высокому уровню, который, в свою очередь, наследует, а при необходимости корректирует или дополняет элементы нижнего уровня.

Для ОССН в иерархическом представлении МРОСЛ ДП-модели выделяют четыре уровня (для СУБД PostgreSQL сформированы отдельные уровни):

- первый (базовый) — модель системы ролевого (дискреционного) управления доступом;
- второй — модель системы ролевого управления доступом и мандатного контроля целостности;
- третий — модель системы ролевого управления доступом, мандатного контроля целостности и мандатного управления доступом только с информационными потоками по памяти;
- четвертый — модель системы ролевого управления доступом, мандатного контроля целостности и мандатного управления доступом с информационными потоками по памяти и по времени.

Аналогично при описании модели на формализованном языке метода Event-B используется техника пошагового уточнения (*refinement*) [14], когда вместо создания монолитной спецификации, уточнение позволяет разрабатывать серию связанных между собой спецификаций, где каждая последующая спецификация в серии является уточнением предыдущих. Таким образом, в формализованной нотации модель также представляется четырьмя соответствующими описанию в математической нотации уровнями уточнений [9]. При этом сам переход на применение пошагового уточнения стал существенным шагом в развитии технологий и практических приемов разработки МРОСЛ ДП-модели и ее дедуктивной верификации.

Вместе с тем детальный анализ реализации механизма управления доступом в программном коде ОССН и добавление в модель в формализованной нотации уровней уточнений, соответствующих механизму управления доступом в СУБД PostgreSQL, показали наличие недостатков и у этих двух рассматриваемых иерархических представлений МРОСЛ ДП-модели. Проанализируем их подробнее.

В ОССН основные механизмы защиты (управление доступом, аудит и др.) реализуются в оригинальной подсистеме безопасности PARSEC [7, 8]. При этом большинство системных вызовов, включающих проверку выполнения правил мандатных управления доступом и контроля целостности, завершаются функцией *pdpm_permission* подсистемы PARSEC, фрагмент которой приведен на рис. 1.

Таким образом, использование единой (для всех случаев проверки выполнения условий мандатного управления доступом и мандатного контроля целостности) функции подсистемы безопасности PARSEC

```

int pdpml_permission(
    const PDPML_T *s, // контекст процесса (субъект-сессии)
    const PDPML_T *o, // контекст файла (сущности)
    int mode // запрашиваемые доступы
)
{
    ...
    // Проверка условий мандатного управления доступом
    if (pdpml_conf_permission(s, o, mode) ) {
        return -EPERM;
    }
    // Проверка условий мандатного контроля целостности
    if (pdp_ilev_permission(s->ilev, o->ilev, mode) ) {
        return -EPERM;
    }
    return 0;
}

```

Рис. 1. Фрагмент кода функции *pdpml_permission*

отвечает лучшим практикам программирования. Она позволяет избегать ошибок, возникающих при многократном повторении одного и того же программного кода, и, при необходимости, предоставляет возможность вносить в него хорошо поддающиеся верификации изменения.

Вместе с тем в МРОСЛ ДП-модели в математической нотации [7], а вслед за ней и в формализованной, следуя сложившимся приемам разработки формальных моделей, функции *pdpml_permission* в предусловиях правил перехода из состояний в состояния моделируемой системы ставились в соответствие проверки нескольких непосредственно не связанных между собой условий.

В качестве примера рассмотрим предусловия и результаты применения предназначенного для описания порядка получения субъект-сессией доступа на запись к сущности правила перехода из состояний в состояния *access_write(x, y)* третьего уровня иерархического представления модели. При этом некоторые элементы описания этого правила, не имеющие прямого отношения к проверке условий мандатных управления доступом и контроля целостности, например, касающиеся порядка предоставления административного доступа на запись к роли, указаны не будут. Одновременно рассмотрим аналогичное этому правилу для случая доступа к сущностям событие *access_write_entity* (рис. 2), включенное в модель в формализованной нотации (соответствующие исходному правилу элементы описания этого события будут указаны в скобках), охранные условия (*grd*) которого задают предусловия события, а действия (*act*) — результаты его применения.

Параметры правила:

x (*subject*) — субъект-сессия, запраивающая доступ на запись;

y (*entity*) — сущность, к которой запрашивается доступ на запись.

Предусловия правила:

$x \in S(\text{grd}1)$ — субъект-сессия принадлежит множеству субъект-сессий;

$y \in E(\text{grd}2)$ — сущность принадлежит множеству сущностей;

существует роль или административная роль $r \in R \cup AR$, принадлежащая множествам ролей или административных ролей (*grd3*);

$(x, r, \text{read}_a) \in AA(\text{grd}3)$ — субъект-сессия *x* имеет к роли или административной роли *r* административный доступ на чтение *read_a* (роль является текущей для субъект-сессии);

$(y, \text{write}_r) \in PA(r)(\text{grd}3)$ — роль или административная роль *r* имеет право доступа на запись *write_r* к сущности *y*;

не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, \text{read}_a) \in AA$ — запрещающая роль *nr* является текущей для субъект-сессии *x*, и $(y, \text{write}_r) \in PA(nr)$ — запрещающая роль *nr* имеет право доступа (запрещающее право доступа) на запись *write_r* к сущности *y* (*grd4*);

$i_e(y) \leq i_s(x)$ — уровень целостности сущности *y* не выше текущего уровня целостности субъект-сессии *x* (*grd7*);

$f_s(x) \leq f_e(y)$ — уровень конфиденциальности сущности *y* не ниже текущего уровня доступа субъект-сессии *x* (*grd12*).

Результаты применения правила:

$A' = A \cup \{(x, y, \text{write}_a)\}$ (*act1*) — во множество текущих доступов добавляется доступ на запись *write_a* субъект-сессии *x* к сущности *y*.

Близкие по составу условия проверяются и в других правилах перехода из состояний в состояния, например, при предоставлении доступа на чтение к сущности, при назначении или удалении роли прав доступа к сущности, при изменении ее роли-владельца, создании, удалении, переименовании сущности. Как следствие, многократное дублирование одинаковых условий создает предпосылки

```

access_write_entity: extended ordinary internal
REFINES
    access_write_entity
ANY
    subject entity
...
WHERE
    grd1: subject ∈ Subjects
    grd2: entity ∈ Entities
    grd3: ∃r · r ∈ OrdRoles ∪ AdmRoles ∧ r ↠ ReadA ∈
        SubjectAdmAccesses(subject) ∧ entity ↠ Write ∈ RoleRights(r)
    grd4: ∀nr · nr ∈ NRoles ∧ nr ↠ ReadA ∈ SubjectAdmAccesses(subject) ⇒
        entity ↠ Write ∉ RoleRights(nr)
...
    grd7: EntityInt(entity) ⊑ SubjectInt(subject)
...
    grd12: SubjectCnf(subject) ⊑ EntityCnf(entity)
THEN
    act1: SubjectAccesses(subject) = SubjectAccesses(subject) ∪
        {entity ↠ WriteA}
END

```

Рис. 2. Фрагмент описания события *access_write_entity*

для ошибок, так как при доработке модели необходимые изменения могут быть внесены не во все такие условия.

Кроме того, при добавлении в МРОСЛ ДП-модель в формализованной нотации уровней уточнений, моделирующих СУБД PostgreSQL, включающую собственный развитый механизм управления доступом и фактически не имеющую с ОССН общих элементов, были выявлены недостатки имеющегося представления модели. Для пояснения этого факта рассмотрим основные использованные в ее формализованной нотации компоненты контекстов (*contexts*) и машин (*machines*) [14]. К их числу относятся: несущие множества (*carrier sets*); константы (*constants*); аксиомы (*axioms* или *axm*); переменные (*variables* или *var*); инварианты (*invariants* или *inv*); события (*events*), включая их параметры (*parameters*), охранные условия (*guards* или *grd*) и действия (*action* или *act*). Для каждого константы, переменной или параметра должен быть определен тип (*type*), являющийся некоторым множеством, которому эта константа, переменная или параметр принадлежит.

В формализованном языке метода Event-B изначально заданы три базовых типа (*basic data types*) [15]:

- \mathbb{Z} — множество целых чисел;
- $BOOL = \{TRUE, FALSE\}$ — булево множество;
- несущие множества (*carrier sets*) — непустые множества, задаваемые в разделе *SETS* контекста.

Над базовыми типами не допускаются операции объединения, пересечения, разности, дополнения, какие-либо дополнительные ограничения на эти мно-

жества могут быть заданы аксиомами. Вместе с тем из базовых типов могут создаваться другие типы, из которых далее также могут создаваться типы. При этом новый тип задается либо как множество всех подмножеств существующего типа (для типа *t* такой тип обозначается через $P(t)$), либо как декартово произведение двух существующих типов (для типов *t1* и *t2* такой тип обозначается через $t1 \times t2$).

При представлении модели в формализованной нотации необходимо в первую очередь определить несущие множества, т. е. основные типы элементов модели. Каждое такое несущее множество будем называть глобальным типом. В модели для первого уровня (ролевого управления доступом) в формализованной нотации таких типов задано четыре (рис. 3):

- *Names* — множество допустимых имен сущностей, ролей, запрещающих и административных ролей;
- *Accesses* — множество видов доступа;
- *AccessRights* — множество видов прав доступа;
- *Union* — множество, включающее все остальные элементы модели (субъект-сессии, сущности, роли и т. д.), которые назовем функциональными элементами модели.

В модели в математической нотации множества прав доступа, видов доступа и допустимых имен сущностей, ролей, запрещающих и административных ролей являются неизменяемыми (множествами-константами). При этом множества субъект-сессий, сущностей и ролей меняются при применении ряда правил перехода из состояний в состояния (например, правила *create_object*).

SETS
Names Accesses AccessRights Union
AXIOMS
UnionIsFinite: finite(Union)
AccessesPartition: partition(Accesses, {ReadA}, {WriteA})
AccessRightsPartition: partition(AccessRights, {Read}, {Write}, {Execute}, {Own})

Рис. 3. Описание глобальных типов

Для моделирования таких правил в машине уровня ролевого управления доступом в формализованной нотации задается переменная $OSUnion \subseteq Union$ (множество существующих перед реализацией соответствующих событий функциональных элементов). При необходимости добавления в событиях новых функциональных элементов считается, что они принадлежат множеству $Union \setminus OSUnion$, т. е. являются "создаваемыми", но еще не "созданными", и имеют тип $Union$.

При включении в модель в формализованной нотации уровней уточнений, соответствующих механизму управления доступом в СУБД PostgreSQL, с одной стороны, для возможности выполнять над множествами функциональных элементов ОССН и СУБД какие-либо общие операции (например, проверять наличие доступов субъект-сессий к сущностям ОССН или СУБД), данные множества должны быть получены из одного глобального типа $Union$. С другой стороны, свойства функциональных элементов ОССН и СУБД сильно отличаются (например, задаваемые к сущностям ОССН или СУБД права доступа или привилегии, соответственно), что затрудняет использование для существующих функциональных элементов СУБД множества $OSUnion$. В результате, эти элементы должны принадлежать множеству $Union \setminus OSUnion$, которое уже используется для моделирования создаваемых элементов ОССН.

Таким образом, анализ приемов формирования МРОСЛ ДП-модели в математической и формализованной нотациях, приведенных в работах [7, 9], позволяет отметить их существенные недостатки. Во-первых, независимое друг от друга описание в правилах перехода из состояний в состояния (или аналогичных им событиях) условий, заданных для выполнения мандатных управлений доступом и контроля целостности, не соответствует реализации этого непосредственно в программном коде ОССН, в котором такая проверка осуществляется только в одной функции *pdml_permission* подсистемы безопасности PARSEC. Во-вторых, многократное дублирование одинаковых условий создает предпосылки для ошибок, когда при корректировке описания модели будут вноситься не все требуемые изменения, что, в свою очередь, затрудняет разработку и верификацию модели, а также ее реализацию в ОССН. В-третьих, технически сложно добавлять в существующее представление модели в формализованной нотации уровни уточнений. Такие уровни моделируют взаимодействующую с ОССН систему (например, СУБД), включающую собственный развитый механизм управления доступом.

Устранение отмеченных недостатков авторами предлагается осуществлять с применением в МРОСЛ ДП-

модели в формализованной нотации тотальных функций и подтипов формализованного языка метода Event-B [14].

Применение тотальных функций

Для устранения первого из отмеченных недостатков в целях логического объединения проверок условий, заданных для выполнения управления доступом моделируемых видов, в МРОСЛ ДП-модели для уровня ролевого управления доступом в математической нотации заданы следующие функции:

- *check_right(s, e, α_r)* — функция наличия прав доступа, значением которой является множество текущих для субъект-сессии s ролей, имеющих право доступа α_r к сущности, роли или субъект-сессии e ;
- *execute_container(s, e)* — функция доступа субъект-сессии к сущностям в контейнерах, значение которой является истинным в случае, когда в иерархии сущностей существует путь к сущности e от некоторой корневой сущности-контейнера, и субъект-сессия s через ее текущие роли имеет право доступа на выполнение *execute*, ко всем сущностям-контейнерам, из которых состоит данный путь, и, наоборот, не имеет запрещающей роли, обладающей правом доступа *execute*, хотя бы к одной сущности-контейнеру этого пути.

В модели в формализованной нотации, представленной в работе [9], данные функции реализовывались несколькими охранными условиями (*grd*) в каждом событии, включая получение доступов к сущностям или ролям, изменение прав доступа ролей и ряде других, составляющих большую часть всех событий в формализованной нотации. Например, на рис. 4 приведено охранное условие *grd5*, реализующее проверку истинности функции *execute_container* в событии получения доступа на запись к сущности *access_write_entity*.

Данное представление функции *execute_container* имеет ряд недостатков, перечисленных далее.

1. Громоздкость и трудочитаемость представления, вследствие чего в нем легко не заметить ошибки. Так, в данном охранном условии проверялось наличие права доступа *Execute* (в математической нотации право доступа на выполнение *execute*) у субъект-сессии не только к контейнерам пути к сущности, но и к самой сущности, что не соответствует определению функции *execute_container* в математической нотации.

2. Повторение этого громоздкого условия в каждом событии, в котором это необходимо, что может дублировать уже допущенную ошибку или внести новую при редактировании условия для конкретного события.

3. Возможное несоответствие охранного условия по сути тому, как аналогичные проверки реализуются в механизме управления доступом ОССН, что должно

```

grd5: ∃E, c · E ⊑ Containers ∧ Root ≠ E ∧ ((entity ∈ dom(EntityNames) ∧
c ∈ dom(EntityNames(entity)) ∧ Parent[E] ∪ {c} = E ∪ {Root}) ∨
(E = ∅ ∧ entity = Root)) ∧ (∀o · o ∈ E ∪ {entity} ∪ {Root} ⇒
(∃r · r ∈ OrdRoles ∪ AdmRoles ∧ r → ReadA ∈
SubjectAdmAccesses(subject) ∧
o → Execute ∈ RoleRights(r)) ∧ (∀nr · nr ∈ NRoles ∧
nr → ReadA ∈ SubjectAdmAccesses(subject) ⇒
o → Execute ≠ RoleRights(nr)))

```

Рис. 4. Реализация функции *execute_container* охранным условием события *access_write_entity*

быть задано некоторым общим для всего представления модели в формализованной нотации инвариантом. Например, при замене в охранном условии права доступа на выполнение *Execute* на право доступа на чтение *Read* (в математической нотации *read*), инструментальным средством Rodin ошибка выдана не будет, так как нигде не было задано, что должно проверяться именно *Execute*, а не другое право доступа.

В связи с перечисленными выше недостатками предлагается подход по представлению функций

check_right и *execute_container* модели в математической нотации в виде тотальных функций (*total function*) *CheckRight* и *ExecuteContainer* в формализованной нотации, состоящих из двух видов антов (рис. 5):

- *инвариант типа*, задающий области определения и значения функции;
- *инвариант истинности*, задающий семантику функции для элементов ее области определения.

Для каждой функции инвариант типа должен быть один, так как области определения и значения

```

CheckRightType:
CheckRight ∈ (Subjects ↔ (Entities ∪ Roles ∪ Subjects ↔
AccessRights)) → P (Roles)
CheckRightFuncE:
∀s,e,ar,r · s ∈ Subjects ∧ e ∈ Entities ∧ ar ∈ AccessRights ∧
r ∈ Roles ⇒ (r ∈ CheckRight({s → {e → ar}})) ⇔
r → ReadA ∈ SubjectAdmAccesses(s) ∧ e → ar ∈ RoleRights(r)
CheckRightFuncR:
∀s,e,ar,r · s ∈ Subjects ∧ e ∈ Roles ∧ ar ∈ AccessRights ∧ r
∈ Roles ⇒
(r ∈ CheckRight({s → {e → ar}})) ⇔ r → ReadA ∈
SubjectAdmAccesses(s) ∧
r ∈ AdmRoles ∧ e → ar ∈ RoleAdmRights(r)
CheckRightFuncS1:
∀s,e,ar,r · s ∈ Subjects ∧ e ∈ Subjects ∧ ar ∈ AccessRights ∧
r ∈ OrdRoles ∪ AdmRoles ⇒ (r ∈ CheckRight({s → {e → ar}})) ⇔ r
→ ReadA ∈
SubjectAdmAccesses(s) ∧ ar = Own ∧ r = SubjectOwner(e)
CheckRightFuncS2:
∀s,e,ar,r · s ∈ Subjects ∧ e ∈ Subjects ∧ ar ∈ AccessRights ∧
r ∈ NRoles ⇒ (r ∈ CheckRight({s → {e → ar}})) ⇔ r → ReadA ∈
SubjectAdmAccesses(s) ∧ ar = Own ∧ r ∈ SubjectNOwners(e)
ExecuteContainerType:
ExecuteContainer ∈ (Subjects ↔ Entities) → BOOL
ExecuteContainerFunc:
∀s,e · s ∈ Subjects ∧ e ∈ Entities ⇒ (ExecuteContainer({s →
e}) = TRUE ⇔
(∃E, c · E ⊑ Containers ∧ Root ≠ E ∧ ((e ∈ dom(EntityNames) ∧
c ∈ dom(EntityNames(e)) ∧ Parent[E] ∪ {c} = E ∪ {Root}) ∨ (E = ∅
∧ e = Root)) ∧
(∀o · o ∈ E ∪ {Root} ⇒ ( ∃r · r ∈ CheckRight({s → {o →
Execute}})) ∧
CheckRight({s → {o → Execute}}) ⊑ OrdRoles ∪ AdmRoles)))

```

Рис. 5. Задание тотальных функций *CheckRight* и *ExecuteContainer*

```

grd3: CheckRight({subject → {entity → Write}}) ⊆ OrdRoles ∪ AdmRoles
grd4: ∃r · r ∈ CheckRight({subject → {entity → Write}})
grd5: ExecuteContainer({subject → entity}) = TRUE

```

Рис. 6. Использование тотальных функций в охранных условиях события *access_write_entity*

задаются однозначно, а инвариантов истинности может быть несколько, поскольку для разных элементов области определения возможны различные условия истинности значения функции. На рис. 6 приведен пример использования тотальных функций в охранных условиях события *access_write_entity*.

Предложенное представление имеет перечисленные далее достоинства.

1. Приближение модели в формализованной нотации к практической реализации механизма управления доступом в ОССН — инварианты истинности тотальных функций можно рассматривать как прототипы спецификаций соответствующих им системных вызовов.

2. Появление четкой связи между инвариантами тотальных функций и событиями модели в формализованной нотации, в охранных условиях которых должны делаться соответствующие проверки.

3. Сигнализирование инструментальным средством Rodin об ошибке при некорректном использовании или переопределении тотальной функции (например, выдача сообщения о недоказуемости сохранности инвариантов для некорректно переопределенной функции в событии).

4. Предотвращение дублирования ошибки или внесения новой при редактировании охранных условий для конкретного события за счет описания громоздких условий истинности в единственной соответствующей тотальной функции.

5. Упрощение задания "контринвариантов" для проверки того, что не доказывается истина из лжи (т. е. в модели в формализованной нотации ошибочно не заданы ложные инварианты или аксиомы).

Для тотальных функций *CheckRight* и *ExecuteContainer* были доказаны все сгенерированные Rodin утверждения (*Proof Obligations — POs*), большинство из которых представляют следующие два вида:

- *INV (invariant preservation)* — для каждого события, изменяющего значение параметра инварианта, требуется доказать, что инвариант остается истинным ("сохраняется") и при новом значении параметра;

- *WD (well-definedness)* — аксиомы, инварианты, охранные условия и действия событий должны быть определены корректно. Например, если в инварианте используется значение тотальной функции для определенного элемента, то будет сгенерировано утверждение для доказательства того, что данный элемент принадлежит области определения данной функции.

Доказательство второго вида *POs* сводится к проверке корректности написания элементов представления модели в формализованной нотации, а именно — аксиом, инвариантов, охранных условий, действий, и выполняется Rodin автоматически.

Для первого вида *POs* доказательство, как правило, значительно сложнее, и для каждой рассматриваемой

тотальной функции оно осуществляется раздельно для инварианта типа и для инвариантов истинности. Для примера проанализируем доказательства утверждений вида *INV* тотальной функции *CheckRight* для события *create_object*. Под гипотезами доказательства далее понимается множество аксиом, инвариантов и охранных условий события, для которого оно проводится, а под гипотезой-целью — цель доказательства.

В соответствующем инварианте типа *CheckRightType* (см. рис. 5) задаются области определения и значения функции с использованием элементов модели в формализованной нотации *Subjects*, *Entities*, *Roles* и глобального типа *AccessRights*, соответствующих множествам субъект-сессий, существ, ролей и видов прав доступа ее в математической нотации. В событиях, модифицирующих эти элементы, для сохранения инварианта необходимо переопределить функцию согласно вносимым изменениям, после чего Rodin автоматически доказывает утверждение *INV* для *CheckRightType*. В рассматриваемом событии *create_object* для этого необходимо добавить создаваемый объект в область определения функции *CheckRight*, выполнив следующие действия (рис. 7).

1. Для переопределения функции *CheckRight* добавить параметр события *checkRight*, который будет соответствовать ее новому значению.

2. В охранном условии (*grd24*) задать тип параметра *checkRight*, т. е. тип новой функции, отличающейся от типа функции *CheckRight* только добавлением к множеству *Entities* создаваемого объекта *object*.

3. В действии (*act8*) заменить функцию *CheckRight* на новую функцию *checkRight*.

Перед тем как проанализировать доказательство для события *create_object* утверждений *INV* для инвариантов истинности функции *CheckRight*, рассмотрим условия сохранения таких инвариантов при изменении в событиях задаваемых ими функций.

При определении некоторой функции *f* ее инвариант истинности в общем случае имеет вид: $y = f(x) \Leftrightarrow P(x, y)$, где x — элемент области определения X ; y — элемент области значения Y (являющийся значением функции при заданном x); $P(x, y)$ — предикат, задающий отношение между элементами X и Y , определяющий *f*. Параметрами данного инварианта являются переменные модели в формализованной нотации, использующиеся в предикате $P(x, y)$. При изменении в событии либо значений параметров инварианта, либо областей определения или значения функции *f* получается новый предикат $P'(x, y)$, который может не равняться исходному предикату (считается, что предикаты $P(x, y)$ и $P'(x, y)$ равны тогда и только тогда, когда совпадают их области определения X и значения Y , и для любых $x \notin X$, $y \notin Y$ таких, что $y = f(x)$, истинно $P(x, y) \Leftrightarrow P'(x, y)$). Таким образом, при доказательстве для событий утверждений *INV*

```

grd24: checkRight ∈ (Subjects ↔ ((Entities ∪ {object}) ∪ Roles ∪
    Subjects ↔ AccessRights)) → P(Roles)

grd25: ∀s,e,ar · s ∈ Subjects ∧ e ∈ Entities ∪ Roles ∪ Subjects ∧
    ar ∈ AccessRights ⇒ checkRight({s ↳ {e ↳ ar}}) =
    CheckRight({s ↳ {e ↳ ar}})

grd26: ∀s,ar,r · s ∈ Subjects ∧ ar ∈ AccessRights ∧ r ∈ Roles ∧ r
    ↳ ReadA ∈ SubjectAdmAccesses(s) ⇒ r ∉ checkRight({s ↳ {object ↳ ar}})

grd27: ∀s,ar,r · s ∈ Subjects ∧ ar ∈ AccessRights ∧ r ∈ Roles ∧ r
    ↳ ReadA ∈
    SubjectAdmAccesses(s) ⇒ (r ∈ checkRight({s ↳ {object ↳ ar}})) ⇔
    object ↳ ar ∈ roleRights(r))

act8: CheckRight = checkRight

```

Рис. 7. Переопределение функции *CheckRight* в событии *create_object*

для инвариантов истинности функций возможны описанные далее два случая.

Первый случай, когда соответствующий инварианту истинности предикат в событии не изменяется. Утверждение *INV* для данного инварианта должно доказываться без дополнительных модификаций события. В процессе доказательства необходимо для заданных элементов областей определения и значения функции проверить истинность нового предиката (гипотезы-цели) с помощью имеющихся гипотез, в числе которых есть исходный предикат. Например, в событии переименования сущности *rename_entity* [9] сохраняется инвариант истинности функции *ExecuteContainer*, поскольку при переименовании сущности (файла или каталога) или входящего в путь до нее контейнера данный путь, как и права доступа к его составляющим, не изменяются. Как следствие, значение функции *ExecuteContainer* не изменяется для всех субъект-сессий и сущностей, и в событии переопределение задающего ее предиката не требуется.

Второй случай, когда в событии изменяется соответствующий инварианту истинности предикат (нарушается хотя бы одно из условий равенства предикатов). В этом случае для сохранения инварианта истинности необходимо переопределить функцию, т. е. задать условия истинности нового предиката, для чего включить в событие охранные условия, задающие предикат для требующих этого параметров, а для остальных — сохранить его исходное отношение. Если переопределение корректно, то утверждение *INV* для инварианта истинности, как правило, доказывается с использованием имеющихся гипотез (аксиом, инвариантов и охранных условий, включая добавленные при модификации события).

Если доказательство провести не удается, то, скорее всего, в переопределении функции допущена ошибка, следовательно, необходимо заново проанализировать логику вносимых изменений и все добавленные элементы модели в формализованной нотации. Но если нет сомнений в корректности переопределения функции и логике математической нотации в целом, а доказать утверждение *INV* не получается, то возможной причиной могут являться ограничения самого инструментального средства Rodin. Например, для функций *CheckRight* и *ExecuteContainer* возникали ситуации, когда средство Rodin не доказывало очевидные гипотезы соответствующих утверждений, а в работе [9] описан случай, когда было невозможно применение математической индукции, для исправления чего потребовалось внесение в контекст первого уровня (ролевого управления доступом) в формализованной нотации хорошо известной в математике аксиомы (рис. 8, где N — множество натуральных чисел, включая ноль).

В связи с этим важно отметить, что если при включении в модель в формализованной нотации дополнительных аксиом или инвариантов, направленных на устранение ограничений Rodin, будет допущена ошибка, то некоторые последующие доказательства, являясь математически ложными, Rodin будут доказаны как истинные. Поэтому такие действия целесообразно осуществлять только в крайних случаях, после проверки того, что с использованием имеющихся гипотез доказательство провести действительно невозможно, а в новых аксиомах или инвариантах не допущены ошибки.

Возвращаясь к рассматриваемому примеру события *create_object* отметим, что при создании но-

```
InductionAxiom: ∀s · s ⊆ N ∧ 0 ∈ s ∧ (∀n · n ∈ s ⇒ n + 1 ∈ s) ⇒ N ⊆ s
```

Рис. 8. Задание аксиомы индукции

вого объекта *object* область определения функции *CheckRight* расширилась. Очевидно, что для всех ее элементов, кроме *object*, значения функции (т. е. предикат для элементов областей определения и значения) не изменяются, так как появление нового объекта не влияет на права доступа к другим существам, ролям или субъект-сессиям. Инвариант истинности должен быть задан на всей новой области определения функции, а значит, для доказательства его сохранения необходимо задать значение функции *CheckRight* для созданного объекта *object*.

Таким образом, переопределение функции *CheckRight* в событии *create_object* должно быть про-должено следующими действиями (см. рис. 7).

1. Для всех субъект-сессий *s*, существ *e* (кроме *object*) и видов прав доступа *ar* соответствующим ох-ранным условием (*grd25*) предикат для новой функ-ции *checkRight* задается его повторением от исходной функции *CheckRight*.

2. Для всех субъект-сессий *s*, видов прав доступа *ar* и созданного объекта *object* задается предикат для функции *checkRight* путем добавления двух ох-ранных условий (*grd26*, *grd27*). В первом условии опре-деляется, что роли, не являющиеся текущими для субъект-сессии *s*, не принадлежат значению функции *checkRight*. Во втором условии — что роли, являю-щиеся текущими для субъект-сессии *s*, принадлежат значению функции *checkRight* тогда и только тогда, когда данные роли содержат право доступа *ar* к *object* (здесь функция *roleRights* является переопределенной, согласно вносимым событием изменениям функцией *RoleRights*, которой в модели в математической нотации соответствует функция прав доступа к сущ-ностям ролей, запрещающих ролей и административных ролей *PA* [9]).

В результате этих действий с помощью Rodin было выполнено доказательство для события *create_object* утверждений *INV* для инвариантов истинности функции *CheckRight*.

Таким образом, задание вместо охранных усло-вий инвариантов тотальных функций *CheckRight* и *ExecuteContainer* привело к тому, что многие вы-полняемые в модели в формализованной нотации до-оказательства стали более содержательными. Они, по сути, соответствуют целям верификации формаль-ной модели, а именно — обоснованию ее коррект-

ности и обеспечению более точной ее реализации в механизме управления доступом ОССН.

Использование подтипов

Для предоставления возможности добавления уровней уточнений, моделирующих взаимодействую-щую с ОССН систему, но не имеющую с ней общих функциональных элементов (в рамках данной рабо-ты — СУБД PostgreSQL), и устранения отмеченного недостатка МРОСЛ ДП-модели в формализованной нотации [9] (основанного на применении только глобальных типов) авторами предлагается исполь-зовать подтипы и их иерархию. При этом подтипа-ми будем называть подмножества глобального типа (для возможности выполнять над ними операции объединения, пересечения, разности, дополнения), для каждой пары которых они либо не пересека-ются, либо один подтип включает другой подтип пары. Иерархией подтипов будем называть заданное на подмножествах некоторого глобального типа *U* бинарное отношение " \leq ", удовлетворяющее условию: для подтипов *A* и *B*, таких что $A \subseteq U$, $B \subseteq U$, $A \subseteq B$, по определению выполняется отношение $A \leq B$.

В результате, на первом уровне (ролевого управ-ления доступом) в формализованной нотации в кон-тексте, представленном на рис. 9, были введены два несравнимых подтипа *OSUnion* (функциональные элемен-ты ОССН) и *DBUnion* (функциональные элемен-ты СУБД). Для этих подтипов выполняются условия: *Union = OSUnion ∪ DBUnion* и *OSUnion ∩ DBUnion = ∅*. В машине введены два моделирующих существующие элементы множества *OSU* и *DBU* (как перменные машины в формализованной нотации), со-ответственно, для ОССН и СУБД такие, что *OSU ⊆ OSUnion* и *DBU ⊆ DBUnion*.

Аналогично были заданы подтипы, соотвеству-ющие учетным записям пользователей (*UsersU*), субъ-ект-сессиям (*SubjectsU*), существам (*EntitiesU*), объектам (*ObjectsU*), контейнерам (*ContainersU*) и ролям (*RolesU*), множества, соотвествующие в том числе существующим объектам (*Objects ⊆ ObjectsU*) и контейнерам (*Containers ⊆ ContainersU*). Это позволило сделать более ясным мо-делирование соотвествующих элементов модели в фор-мализованной нотации и расширить возможности Rodin по проверке корректности их использования.

```

SETS
    Union
CONSTANTS
    OSUnion DBUnion UsersU SubjectsU EntitiesU ObjectsU ContainersU RolesU
AXIOMS
    UnionPartition: partition(Union, OSUnion, DBUnion)
    OSUnionPartition: partition(OSUnion, UsersU, SubjectsU, EntitiesU,
        RolesU)
    EntitiesUPartition: partition(EntitiesU, ObjectsU, ContainersU)

```

Рис. 9. Иерархия подтипов модели в формализованной нотации для уровня ролевого управления доступом ОССН

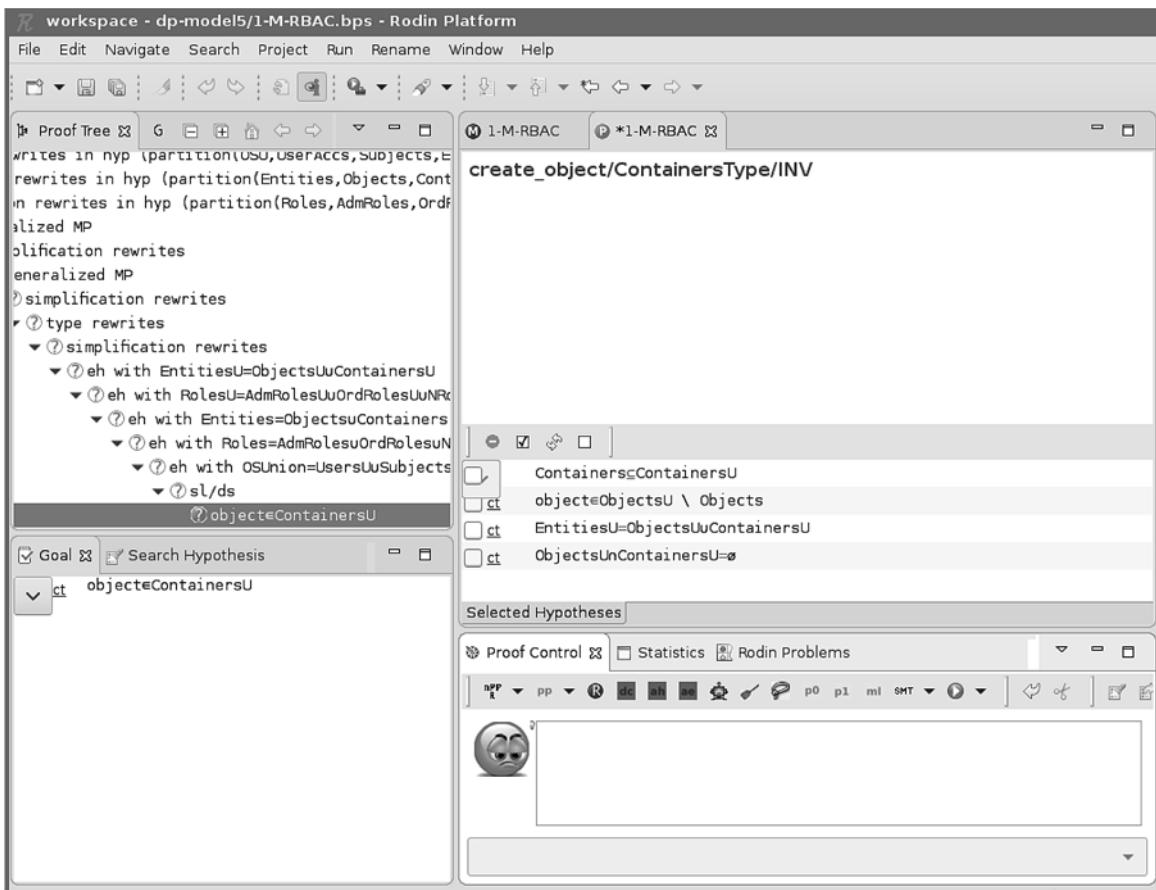


Рис. 10. Результат попытки доказательства Rodin ложной гипотезы-цели

Например, при добавлении объекта ОССН *object* он имеет подтипы $ObjectsU \subseteq EntitiesU \subseteq OSUnion \subseteq Union$ и принадлежит множеству $OSUnion \setminus OSU$. Это означает, что такой объект заведомо не принадлежит множеству $DBUnion$. Кроме того, *object* принадлежит множеству $ObjectsU \setminus Objects$, т. е. сразу указывается на его принадлежность объектам ОССН. Это позволяет избежать ошибки, например, добавления *object* во множество контейнеров *Containers*, что было бы возможно, если бы был задан только подтип *OSUnion*. Теперь при добавлении объекта с подтипом *ObjectsU* во множество *Containers* будет невозможно доказать утверждение *INV* с ложной гипотезой-целью " $Containers \cup \{object\} \subseteq ContainersU$ ", так как в контексте задана аксиома *EntitiesUPartition*, из которой следует, что $ContainersU \cap ObjectsU = \emptyset$ (рис. 10).

Представление глобального типа в виде объединения непересекающихся подтипов позволяет сделать модель в формализованной нотации более структурированной и приспособленной к добавлению новых уровней уточнения или модификации существующих. Причина в том, что при необходимости задать новый подтип его добавление не повлияет на все элементы с большим или несравнимым подтипов в их иерархии. Это свойство показано на примере добавления уровня уточнения для ролевого управления доступом СУБД PostgreSQL, когда был добавлен новый подтип, являющийся подмножеством *Union* и не имеющий пересечений с подтипом *OSUnion*.

Заключение

На основе анализа приведенных в работе [9] представлений МРОСЛ ДП-модели в математической и формализованной нотациях были отмечены их недостатки. К их числу относятся: независимое друг от друга описание элементов модели для различных видов управления доступом; многократное дублирование одинаковых условий их выполнения, не соответствующее реализации проверок этих условий непосредственно в программном коде ОССН; сложность добавления уровней уточнений, моделирующих системы, взаимодействующие с ОССН. Для устранения этих недостатков, которые наиболее существенны для модели в формализованной нотации, предложено: логически объединить проверки условий, заданных для мандатных управления доступом, контроля целостности и ролевого управления доступом; улучшить структуру использованных при этом элементов модели, для чего применять тотальные функции и подтипы формализованного языка метода Event-B.

Результатом использования перечисленных приемов при доработке МРОСЛ ДП-модели стало ее согласованное описание в формализованной и математической нотациях. Такое описание способствовало, во-первых, более адекватной практической реализации механизма управления доступом в ОССН Astra Linux Special Edition, во-вторых, упрощению дедук-

тивной верификации модели в формализованной нотации с применением инструментального средства Rodin, а также автоматизированному выявлению при этом возможных ошибок или их дублирования, в-третьих, снижению затрат усилий по добавлению уровней уточнений, моделирующих системы, взаимодействующие с ОССН и самостоятельно реализующие развитые механизмы управления доступом, например, СУБД PostgreSQL. Предложенные в работе приемы используются при развитии самой МРОСЛ ДП-модели. Они также могут быть полезны при разработке других формальных моделей управления доступом и их верификации с применением инструментальных средств, что отвечает актуальным нормативным документам ФСТЭК России [12].

Список литературы

1. Bell D. E., LaPadula L. J. Secure Computer Systems: Unified Exposition and Multics Interpretation. Bedford, Mass.: MITRE Corp., 1976. MTR-2997 Rev. I. — 129 p.
2. Девянин П. Н. Модели безопасности компьютерных систем. Управление доступом и информационными потоками. Учебное пособие для вузов. 2-е изд., испр. и доп. — М.: Горячая линия — Телеком, 2013. — 338 с.
3. Biba K. J. Integrity Considerations for Secure Computer Systems. Bedford, Mass.: MITRE Corp., 1975. MTR-3153. — 61 p.
4. Bradley T. Introduction to Windows Integrity Control. URL: <http://www.symantec.com/connect/articles/introduction-windows-integrity-control>
5. Sandhu R. The typed access matrix model // In Proceeding of the IEEE Symposium on Research in Security and Privacy. — Oakland, CA, May 1992. — P. 122—136.
6. Security-Enhanced Linux. URL: <https://www.nsa.gov/what-we-do/research/selinux/>
7. Буренин П. В., Девянин П. Н., Лебеденко Е. В. и др. Безопасность операционной системы специального назначения Astra Linux Special Edition.: Учеб. пособие для вузов / Под ред. П. Н. Девянина. 3-е изд. перераб. и доп. — М.: Горячая линия — Телеком, 2019. — 404 с.
8. Astra Linux — Универсальная операционная система. URL: <http://www.astralinux.ru>
9. Девянин П. Н., Ефремов Д. В., Кулямин В. В. и др. Моделирование и верификация политик безопасности управления доступом в операционных системах. — М.: Горячая линия — Телеком, 2019. — 214 с.
10. Девянин П. Н. Уровень запрещающих ролей иерархического представления МРОСЛ ДП-модели // Прикладная дискретная математика. — 2018. — № 1 (39). — С. 58—71.
11. Девянин П. Н. Подходы к моделированию управления доступом в СУБД PostgreSQL в рамках МРОСЛ ДП-модели // Прикладная дискретная математика. — 2018. — Приложение № 11. — С. 95—98.
12. Информационное сообщение ФСТЭК России от 29.03.2019 № 240/24/1525. URL: <https://fstec.ru/component/attachments/download/2286>.
13. Девянин П. Н. О результатах формирования иерархического представления МРОСЛ ДП-модели // Прикладная дискретная математика. — 2016. — Приложение № 9. — С. 83—87.
14. Abrial J.-R., Hallersteide S. Refinement, decomposition, and instantiation of discrete models: Application to Event-B // Fundamenta Informaticae. — 2007. — Vol. 77, Issue 1—2. — P. 1—28.
15. Rodin Handbook. URL: <https://www3.hhu.de/stups/handbook/rodin/>

Application of Subtypes and Total Functions of Event-B Formal Method for the Formalization and Verification of the MROSL DP-Model

P. N. Devyanin, pdevyanin@astralinux.ru, M. A. Leonova, mleonova@astralinux.ru, RusBITech-Astra, Moscow, 117105, Russian Federation

Corresponding author:

Devyanin Petr N., Chief Researcher, RusBITech-Astra, Moscow, 117105, Russian Federation
E-mail: pdevyanin@astralinux.ru

Received on March 25, 2020
Accepted on May 20, 2020

Formal access control models are used to design, implement and justify the security of the access control mechanism, which is part of many information security systems such as operating systems (OS), DBMS. Russian practice of developing information security systems, it is advisable to mention the mandatory entity-role model of access control and information flows in OS of Linux set (MROSL DP-model), developed for implementation in OS Astra Linux Special Edition. The model is formalized using Event-B formal method (formalized notation) and its correctness is formally verified with the help of the Rodin toolset. This article presents methods of formalization of the MROSL DP-model using subtypes and total functions of Event-B, which allow logically integrate checks of conditions specified for role-based access control (RBAC), mandatory integrity control (MIC) and multi-level security (MLS), and improve the structure of formalized notation elements used. The result of using these methods in refining the formalized notation was its approach to the practical implementation of the access control mechanism in the OS Astra Linux Special Edition; simplification of formal deductive verification of formalized notation using the Rodin tool, at the same time automated detection of possible errors or their duplication, addition of refinement levels also. Refinement levels simulate (other) systems interacting with OS. Those systems implement complex access control mechanisms independently, for example, DBMS PostgreSQL. These methods are used in development of the MROSL DP-model, and can also

be useful in development of other formal access control models and verification using tools, which meet the actual regulatory documents of FSTEC Russia.

Keywords: formal access control model, verification, Event-B, operating system

For citation:

Devyanin P. N., Leonova M. A. Application of Subtypes and Total Functions of Event-B Formal Method for the Formalization and Verification of the MROSL DP-Model, *Programmnaya Ingeneria*, 2020, vol. 11, no. 4, pp. 230–241

DOI: 10.17587/prin.11.230-241

References

1. Bell D. E., LaPadula L. J. *Secure Computer Systems: Unified Exposition and Multics Interpretation*, Bedford, Mass. MITRE Corp., 1976, MTR-2997 Rev. 1, 129 p.
2. Devyanin P. N. *The models of security of computer systems: access control and information flows*, Moscow, Hot line — Telecom, 2013, 338 p. (in Russian).
3. Biba K. J. Integrity Considerations for Secure Computer Systems, Bedford, Mass.: MITRE Corp., 1975, MTR-3153. 61 p.
4. Bradley T. Introduction to Windows Integrity Control, available at: <http://www.symantec.com/connect/articles/introduction-windows-integrity-control>
5. Sandhu R. The typed access matrix model, In *Proceeding of the IEEE Symposium on Research in Security and Privacy*, Oakland, CA, May 1992, pp. 122–136.
6. Security-Enhanced Linux, available at: <https://www.nsa.gov/what-we-do/research/selinux>.
7. Burenin P. V., Devyanin P. N., Lebedenko E. V., Proskurin V. G., Tsibulya A. N. *Security of operating system Astra Linux Special Edition*, Moscow, Hot-line Telecom, 2019, 404 p. (in Russian).
8. Astra Linux — Universal operating system, available at: <http://www.astralinux.ru>
9. Devyanin P. N., Efremov D. V., Kuliamin V. V., Petrenko A. K., Khoroshilov A. V., Shchepetkov I. V. *Modeling and verification access control security policies on operating systems*, Moscow, Hot-line Telecom, 2019, 214 p. (in Russian).
10. Devyanin P. N. The level of negative roles of the hierarchical representation of MROSL DP-model, *Prikladnaya diskretnaya matematika*, 2018, no. 1(39), pp. 58–71 (in Russian).
11. Devyanin P. N. Approaches to formal modeling of access control in PostgreSQL within framework of the MROSL DP-model, *Prikladnaya diskretnaya matematika*, 2018, supplement no. 11, pp. 95–98 (in Russian).
12. Information message of FSTEC Russia dated 29.03.2019 № 240/24/1525, available at: <https://fstec.ru/component/attachments/download/2286> (in Russian).
13. Devyanin P. N. About results of design hierarchical representation of MROSL DP-model, *Prikladnaya diskretnaya matematika*, 2016, supplement no. 9, pp. 83–87 (in Russian).
14. Abrial J.-R., Hallerstede S. Refinement, decomposition, and instantiation of discrete models: Application to Event-B, *Fundamenta Informaticae*, 2007, vol. 77, issue 1–2, pp. 1–28.
15. Rodin Handbook, available at: <https://www3.hhu.de/stups/handbook/rodin/>

ИНФОРМАЦИЯ

IV Международная конференция «УСТОЙЧИВОСТЬ И ПРОЦЕССЫ УПРАВЛЕНИЯ»

Международная конференция «Устойчивость и процессы управления», посвященная 90-летию со дня рождения профессора, чл.-корр. РАН В. И. Зубова (1930—2000), будет проходить 5—9 октября 2020 г. в Санкт-Петербурге на базе факультета прикладной математики—процессов управления Санкт-Петербургского государственного университета (СПбГУ).

Научные направления конференции

- ◆ Устойчивость
- ◆ Метод функций Ляпунова
- ◆ Теория динамических систем
- ◆ Управление механическими системами
- ◆ Управление и оптимизация в электрофизических системах
- ◆ Управление конфликтными системами.
Динамические игры
- ◆ Методы анализа и синтеза систем с последействием
- ◆ Робастность
- ◆ Методы оптимизации
- ◆ Нелинейная механика и физика твердого тела
- ◆ Управление социально-экономическими системами
- ◆ Управление медико-биологическими системами
- ◆ Информатика и процессы управления
- ◆ Математические проблемы и методы распознавания образов
- ◆ Искусственный интеллект

Контактная информация: Егоров Алексей Валерьевич, alexey.egorov@spbu.ru

Головкина Анна Геннадьевна, a.golovkina@spbu.ru

Официальный сайт конференции <http://www.apmath.spbu.ru/scp2020/>

М. А. Черепнёв, д-р физ.-мат. наук, доц., cherepniov@gmail.com,
МГУ имени М. В. Ломоносова

Оценка скорости работы нового параллельного блочного алгоритма для решения задач в области больших разреженных систем над большим простым полем*

Рассмотрен вопрос о том, как оценить время работы программы, реализующей алгоритм нахождения решения большой разреженной системы линейных уравнений над большим простым полем, изложенный в работах [1, 2], на вычислительной системе из K независимых вычислительных узлов. Блочный характер предлагаемого алгоритма позволит уменьшить время работы последовательной скалярной версии (алгоритм Монтгомери [3]) примерно в \sqrt{K} раз, при K , растущем до размеров исходной задачи. Общее время работы всей программы нахождения решения разреженной системы линейных уравнений в пространстве размерности N снижено до $O\left(\frac{N^2}{\sqrt{K}}\right)$ при произвольном K , от 1 до N . Таким образом, теоретически получено оптимальное время $O(N^{3/2})$.

Ключевые слова: дискретное логарифмирование, метод Ланцоша, параллельные вычисления, блочные алгоритмы

Введение

Задача нахождения некоторого количества (не обязательно всех) решений большой разреженной системы уравнений над большим простым полем возникает при решении задачи дискретного логарифмирования, которая связана со стойкостью очень многих систем защиты информации. Похожие исследования для случая GF(2) описаны в работе [4]. Однако в настоящей статье предлагается ряд принципиально новых приемов, позволяющих существенно уменьшить время работы программы. Таким образом, настоящая работа имеет прежде всего важное прикладное значение. Цель работы — минимизация времени за счет увеличения числа используемых вычислительных узлов.

Рассмотрена топология, связывающая узлы вычислительной сети типа "тор". При этом считается, что в каждом из двух направлений этого тора действуют связи с разной пропускной способностью. Реально это может соответствовать крупным вычислительным кластерам, связанным относительно медленной сетью Интернет, либо нескольким многоядерным (возможно, с виртуальными ядрами) процессорам одного кластера. При этом в оценках времени работы учитывается не только время, затрачиваемое на счет, как это сделано в работе [2], но и время на пересылку. Около половины всех коммутаций допускается медленными, откуда получена возможность набрать необходимое количество вы-

числительных узлов из разных кластеров. Предложено специфическое распределенное хранение промежуточных результатов вычислений, в итоге получена экономия не только вычислительных ресурсов, но и времени на передачу, которая осуществляется параллельно по независимым ветвям заданной сети. Кроме того, решение плотных систем линейных уравнений, которое необходимо делать на каждом шаге представленного алгоритма, а также обращения плотных матриц предлагается осуществлять с помощью параллельных вычислений. Это позволит снизить время обращения матриц размера $s \times s$ до $O(s^2)$, а время работы последовательной скалярной версии (алгоритм Монтгомери [3]) примерно в \sqrt{K} раз, при K , растущем до размеров исходной задачи. Общее время работы всей программы решения разреженной системы линейных уравнений в пространстве размерности N до N^2/\sqrt{K} при произвольном K , от 1 до N . Таким образом, теоретически получено оптимальное время $O(N^{3/2})$.

Основные операции, нуждающиеся в распараллеливании, это умножение разреженной матрицы на блок, умножение этих блоков друг на друга (скалярное произведение) и на относительно маленькие матрицы (линейные комбинации), а также решение линейных систем с матрицами относительно маленького, но все же достаточно большого размера.

Каждую операцию любого алгоритма можно рассматривать на предмет возможности использования нескольких однотипных узлов для уменьшения времени на ее реализацию. При этом если считать число арифметических операций в рассматриваемом

* Работа поддержана грантом РФФИ 18-29-03124 мк.

алгоритме фиксированным, то самое значительное снижение времени — это падение времени обратно-пропорционально числу вычислительных узлов. Так будет, например, при наборе соотношений в методах дискретного логарифмирования и факторизации на основе факторных баз. Однако для большинства относительно более сложных алгоритмов верхняя оценка времени их работы в зависимости от параметров задачи и используемого оборудования представляется как $T(N, d, n, c...; s)$, где $N, d, n, c...$, будут введены позже, и числа используемых вычислительных узлов s . Эта оценка может вести себя по-разному на разных областях изменения аргумента. Зачастую, при увеличении s время не падает обратнопропорционально s . А при увеличении s выше определенного порогового значения $s_0(N, d, n, c...)$, зависящего от параметров задачи, оно снова начинает увеличиваться вследствие того, что время на обмен между s вычислительными узлами растет быстрее, чем уменьшается время работы каждого вычислительного узла в отдельности.

По мнению автора, правомерно поставить вопрос о вычислении значения величины $s_0(N, d, n, c...)$ и $T(N, d, n, c...) = \min_s T(N, d, c...; s) = T(N, d, n, c...; s_0(N, d, n, c...))$ для некоторого эталонного кластера. В качестве эталона на настоящее время логично взять кластер с неограниченным числом вычислительных узлов, для которых время выполнения одной арифметической операции с машинными словами, умноженное на некоторую константу c , равно времени передачи одного машинного слова между вычислительными узлами. В современной компьютерной технике обмен между оперативной памятью и процессором осуществляется с $c \approx 5$, а между отдельными частями оперативной памяти — $c \approx 20$ (время на обмен с кешем процессора в этой статье учитываться не будет).

Несколько известно автору, параметр c определяется соотношением между тактовыми частотами процессора и шины, которая связывает вычислительные узлы, процентом информационных битов в передаваемых по внутренней сети сообщениях и некоторыми другими характеристиками кластера. Будем также предполагать, что на кластере возможна как адресная рассылка между выполняющими конкретные задания вычислительными узлами, так и рассылка с фиксированного узла на узлы некоторой группы по бинарному дереву (брюдкаст), т. е. за логарифм от числа элементов этой группы. Пересылки внутри непересекающихся групп вычислительных узлов будем считать независимыми. Все оценки времени проведены в единицах, равных времени на выполнение одной арифметической операции с машинными словами в таком эталонном кластере. Будем считать также незначительным время на преобразование форматов хранения матриц. Все дальнейшие результаты получены при указанных условиях.

Задачу решения большой разреженной системы линейных уравнений

$$AX = B, B; X \in \mathbb{F}^{N \times s},$$

будем характеризовать четырьмя параметрами:

$N \geq 2^{20}$ — размер исходной матрицы (максимум из числа строк и столбцов); d — оценка числа ненулевых

элементов в каждой строке этой матрицы; s — параметр, о котором было сказано выше. Таким образом, пересылка V бит будет осуществлена за $\frac{c}{n}V$ единиц времени, где n — длина машинного слова. Одна единица времени — это время, необходимое для выполнения одной арифметической операции с машинными словами. Параметр s принято называть блочным фактором. В дальнейшем вместо векторов длины N будем рассматривать блоки, состоящие из s столбцов, а вместо скаляров — матрицы размера $s \times s$.

Решение ищем в пространстве Крылова, а именно в линейной оболочке столбцов матриц $\langle B, AB, A^2B, \dots \rangle$. Это решение может быть построено по формуле

$$X = \sum_{i=1}^{N/s} W_i (W_i^\top A W_i)^{-1} W_i^\top B,$$

где $\langle B, AB, A^2B, \dots \rangle = \langle W_0, W_1, \dots \rangle$, а матрицы W_i образуют базис пространства Крылова, ортогональный относительно скалярного произведения с матрицей A , и обладают некоторыми дополнительными свойствами. При этом матрицу A можно считать симметричной и вырожденной [1, 5].

В настоящей работе предпочтение отдано блочной версии ($s > 1$), поскольку при распараллеливании скалярной версии на первый план выходит время, затрачиваемое на пересылки. Например, чтобы собрать с помощью алгоритма циклической пересылки [6] вектор размера N , потребуется время порядка $O(N)$, а таких векторов в пространстве Крылова также N . Таким образом, получается время порядка N . Блочный подход позволяет распараллелить самую трудную часть алгоритма — построение пространства Крылова, используя тот факт, что после каждого умножения блока на матрицу результат не надо снова собирать в блок. Следует отметить, что в блочном подходе есть свой принципиальный минус — это растущие как квадрат блочного размера вычислительные затраты на скалярное произведение, при том, что число необходимых скалярных произведений уменьшается пропорционально его первой степени. Поэтому приходится увеличивать число вычислительных узлов, чтобы выполнить эту работу. Именно поэтому эффективность использования вычислительных узлов в этом алгоритме определяется знаменателем вида \sqrt{K} . Однако, поскольку K может расти с сохранением этой оценки примерно до N , существует возможность посчитать решение линейной системы за время порядка $N^{3/2}$, что невозможно на сегодняшний момент сделать другим алгоритмом на вычислительной системе такого же размера.

Рассмотрим следующую вычислительную систему, состоящую из $ls + t = K$ однотипных вычислительных узлов:

$$\left| \begin{array}{cccccc|c} \circ & \circ & \cdots & \circ & & & t \\ \circ & \circ & \cdots & \circ & & & \vdots \\ \vdots & \vdots & & \vdots & & \cdots & \\ \circ & \circ & \cdots & \circ & & & \end{array} \right| s$$

Такая конструкция является общей для данного метода. В настоящей работе эта конструкция рас-

сматривается при условии, что вертикальные связи между узлами быстрее горизонтальных. Коэффициенты, соответствующие коэффициенту c , который упоминался выше, равны c_1 для вертикальных и c_2 для горизонтальных связей, $c_1 < c_2$. Левая часть этой вычислительной системы, состоящая из $s l$ вычислительных узлов, будет вычислять куски пространства Крылова, необходимые скалярные произведения, а также решение исходной системы. А правая часть, состоящая из t узлов, будет вычислять коэффициенты этого решения. Далее фактически будет написана программа (алгоритм работы) для каждого из однотипных вычислителей правой и левой частей вычислительной системы, осуществляющая баланс времени на счет и пересылки. Конечно, функции правой части вычислительной системы можно передать узлам левой части, сэкономив при этом время на пересылках. Однако в этом случае каждый узел должен будет проводить вычисления элементов системы одновременно по нескольким программам. Для некоторых (многоядерных) вычислителей это может и не привести к дополнительным затратам времени. Однако в контексте целей настоящей работы будем следить за тем, чтобы все вычислительные ресурсы каждого узла были постоянно задействованы в вычислениях, и на работу параллельной программы уже было нечего выделять.

Параллельное решение линейных систем

Как было отмечено выше, любую процедуру можно раздать на несколько вычислительных узлов и тем самым уменьшить время на вычисления в такое же число раз. Например, для приведения матрицы размера $s \times s$ к треугольному виду или для получения ее LU-разложения требуется не более s^3 операций. Значит на системе из s вычислительных узлов это потребует не более s^2 операций на каждом вычислительном узле. Однако для осуществления этих операций каждому вычислительному узлу может потребоваться s или больше элементов, вычисленных на других узлах, и ускорения не получится. Но если распараллеливать приведение к треугольному виду по столбцам, то для реализации каждого элементарного преобразования узлам необходимо получить всего один элемент и сделать одно умножение и одно вычитание. Поскольку необходимых элементарных преобразований не более $s^2/2$, получаем оценку времени для всего алгоритма приведения к треугольному виду $c_1 s^2/2$. Для решения системы, имеющей треугольный вид, требуется не более $s^2/2$ операций. Это означает, если каждому из s вычислительных узлов поручить в качестве правой части свой базисный вектор, то время на обращение матрицы потребуется в сумме не более $c_1 s^2$ с учетом рассылки всей треугольной матрицы. Аналогичное распараллеливание можно предложить и для получения LU-разложения. Для этого достаточно раздать s вычислительным узлам s -ю долю элементов матрицы для вычисления, равномерно разбросав их по всей матрице для балансировки нагрузки.

Формальное описание алгоритма

Будем пользоваться обозначениями, введенными в работе [1], только для случая большого простого поля F . Алгоритм работает в виде цикла, который строит последовательно часть ортогонального базиса пространства Крылова и соответствующие координаты решения.

При входе в очередной цикл алгоритма имеем k старших коэффициентов многочленов — матричных приближений Паде $Q^{(k)}(\lambda)$, $Q^{(k-1)}(\lambda)$ к ряду $\alpha(\lambda)$. Сначала на левой части вычислительной системы вычисляем блочные векторы

$$A^i Q^{(k)}(A, B), \quad A^i Q^{(k-1)}(A, B), \quad i = 0, 1, \dots, k, \quad (1)$$

где k — параметр, который будет выбран ниже. Эти векторы, также как и вектор B , будут иметь s столбцов, которые в соответствии с их номерами при всех $i = 0, 1, \dots, k$ хранятся на соответствующих группах вычислительных узлов левой вычислительной системы. Внутри группы будем хранить их распределено по высоте на каждом по $\frac{N}{l}$ элементов каждого столбца.

Кроме того, в результате предыдущей синхронизации вычислены начальный отрезок решения (до k -го слагаемого включительно) и матрицы $W_i^T B$ при $i = k, k+1$, а также $(W_k^T A W_k)^{-1}$.

Вычислим коэффициенты γ_i и μ_i по следующим формулам:

$$\begin{aligned} (Q^{(k-1)}(A, B))^T A^j Q^{(k-1)}(A, B) &= \sum_{i=k}^{k+j-1} (Q_{i-j}^{(k-1)})^T \gamma_i, \\ (Q^{(k)}(A, B))^T A^j Q^{(k)}(A, B) &= \sum_{i=k+1}^{k+j} (Q_{i-j}^{(k)})^T \mu_i. \end{aligned} \quad (2)$$

Не дожидаясь окончания этих вычислений, запускаем циклический процесс, состоящий из повторяющихся двух частей, описанных ниже. Обозначим коэффициенты $Q_{j,i}^{(t)}$ в соответствии с формулами

$$Q^{(t)}(\lambda) = \sum_{i=0}^{t-k} \lambda^i Q^{(k)}(\lambda) Q_{k,i}^{(t)} + \sum_{i=0}^{t-k-1} \lambda^i Q^{(k-1)}(\lambda) Q_{k-1,i}^{(t)}, \quad t \geq k+1. \quad (3)$$

Начальные условия определим так: $Q_{k,0}^{(k-1)} = O_n$, $Q_{k-1,0}^{(k-1)} = I_n$, $Q_{k,0}^{(k)} = I_n$, $Q_{k-1,0}^{(k)} = O_n$ (остальные коэффициенты нулевые).

Для получения $Q_{j,i}^{(t+1)}$, сначала по формулам

$$\begin{aligned} \tau_j &= \sum_{i=0}^{t-k-1} \mu_{i+j} Q_{k,i}^{(t-1)} + \sum_{i=0}^{t-k-2} \gamma_{i+j} Q_{k-1,i}^{(t-1)}, \\ \tau_j &= \mu_j \text{ для } t = k+1, \\ \rho_j &= \sum_{i=0}^{t-k} \mu_{i+j} Q_{k,i}^{(t)} + \sum_{i=0}^{t-k-1} \gamma_{i+j} Q_{k-1,i}^{(t)}. \end{aligned} \quad (4)$$

вычисляются соответствующие коэффициенты левых частей в очередных формулах

Вертикальный размер этой матрицы равен $2k + 3$ блокам. Отметим, что при $t = k + 1$ строки, отмеченные подчеркиванием, следует опустить, а элементы той же матрицы, помеченные точками, заменить нулевыми матрицами.

Решая однородную систему в верхних двух блочных строках, получаем коэффициенты рекуррентных формул v_i , а затем коэффициенты очередных приближений в текущем базисе. Указанные коэффициенты снова используются в формулах (4) на следующих шагах.

Для блочных векторов длины $2k + 1$, расположенных в этих равенствах справа, начиная с третьей позиции сверху, введем следующие обозначения:

На выходе из этой части:

- рекуррентные формулы (т. е. коэффициенты v_i);
 - коэффициенты вектора $Q^{(t+1)}$;
 - в начале следующего цикла находим старший коэффициент $Q_{t+1}^{(t+1)}$, вычисленный по формулам (3) для вычисления скалярных произведений $W_{t+1}^T A W_{t+1} = Q_{t+1}^{(t+1)T} \sigma_{t+2}$, где σ_i — коэффициенты правых частей Паде разложений:

$$\alpha(\lambda)Q^{(t+1)}(\lambda) - P^{(t+1)}(\lambda) = \sum_{i=t+2}^{\infty} \sigma_i \lambda^{-i},$$

которые вычисляют по формулам (4).

Вертикальный размер в формуле (5) соответствует максимальному числу коэффициентов в формуле (3) плюс две блочных строки на коэффициенты Паде разложений, которые надо обнулить на текущем шаге.

Обработка координат

С помощью рекуррентных формул последовательно в правой части вычислительной системы проводятся следующие действия.

1. Построение $\widehat{W}_{t+1} = AW_t + \widehat{W}_{tV_0} + \widehat{W}_{t-1V_{-1}}$, здесь $\widehat{W}_t = Q^{(t)}$, а $\widehat{A}W_t$ обозначает вектор, ненулевые координаты которого в текущем базисе представляют собой свинутые на одну позицию вверх координаты вектора \widehat{W}_t . На первых двух шагах алгоритма вычисляем элементы напрямую. Полученный новый блок

по построению будет ортогонален всем предыдущим. Заметим, что $W_t = \hat{Q}^{(t)}$ означает, что доортогонализация, необходимая в работе [1], в данном случае не нужна, так как все матрицы размера $s \times s$ считаем невырожденными, ввиду очень большой вероятности этого в большом простом поле.

2. Вычисление $(W_{t+1}^T A W_{t+1})^{-1}$ с помощью параллельного алгоритма и решения s линейных систем с правыми частями в виде базисных единичных векторов.

3. Вычисление $W_{t+1}^T B = v_0^T W_t^T B + v_{-1}^T W_{t-1}^T B$, здесь используется то, что $\hat{W}_t^T AB = 0$ по построению, а на первых двух шагах алгоритма вычисления проводятся напрямую.

Элементы $(W_i^T A W_i)^{-1}, W_i, W_i^T B$ запоминаются на правой части вычислительной системы. Затем проводится **синхронизация**, суть которой заключается в описанных далее действиях.

Считаем, что матрица $Q_{k,k}^{(2k)}$, а также матрица τ_{2k} в представлении

$$\alpha(\lambda) Q^{(2k-1)}(\lambda) - P^{(2k-1)}(\lambda) = \sum_{i=2k}^{\infty} \tau_i \lambda^{-i}, \quad (6)$$

которые вычисляются по формулам (2), (4) — невырождены.

На самом деле в этой версии алгоритма на каждом шаге нужна невырожденность старших коэффициентов разложения в правых частях равенств (6) для решения линейных систем в верхних двух блочных строках системы (5), будем ее предполагать, ввиду большой вероятности этого события при работе в большом простом поле.

Перевычисляется решение

$$X = X + A_1 \sum_{i=k+1}^{2k} \hat{W}_i (W_i^T A W_i)^{-1} W_i^T B = X + A_1 \hat{X}, \quad (7)$$

где матрица A_1 состоит из блочных векторов (1). Для этого слагаемые правой суммы вычисляются на правой части вычислительной системы и передаются в левую часть построчно, каждая строка к столбцу с тем же номером. Сложение происходит по методу циклической пересылки по медленным горизонтальным каналам.

Аналогично строятся новые образующие блоки по формулам

$$Q^{(i)}(A, B) = A_1 \hat{Q}^{(i)}, i = 2k-1, 2k, \quad (8)$$

с последующей раздачей результата по столбцам на s вычислительных узлов левой части вычислительной системы.

По формулам (3) на правой части вычислительной системы при $t = 2k, 2k-1$ вычисляются k старших коэффициентов многочленов $Q^{(2k)}(\lambda), Q^{(2k-1)}(\lambda)$ из k старших коэффициентов многочленов предыдущих образующих для вычислений по формулам (2).

Далее на левой части вычислительной системы вычисляются левые части формул (2). Для этого на каждом шаге после вычисления очередного вектора с номером i по формуле (1) узлами фиксированного столбца проводят умножение на него слева матрицами $(Q^{(k-1)}(A, B))^T, (Q^{(k)}(A, B))^T$, которые хранятся на них распределенно по вертикали на l одинаковых частей.

Рассмотренная версия алгоритма, основы которого были предложены в работе [1], делает практически те же действия, что и алгоритм Видемана—Копперсмита [7], но частями для сравнительно небольших отрезков базиса пространства Крылова. Это позволяет не проводить повторного вычисления, а запоминать. Уже использованные для построения решения отрезки этого базиса забываются. Но важно то, что это позволяет избежать работы с многочленами большой степени, как следствие, приводит не только к экономии памяти, но и к увеличению скорости работы. В качестве платы за эти дополнительные удобства выступает необходимость периодически проводить "синхронизацию", т. е. фактически собирать результат умножения матрицы на блок. Однако оптимальное число необходимых синхронизаций, как будет показано ниже, невелико, и для текущих размеров задач составляет чуть больше 100.

Работа левой части вычислительной системы

Прежде чем перейти к формулировкам теорем, отметим следующее: в последовательных реализациях указанных алгоритмов максимальное время тратится на многократно повторяющуюся операцию умножения матрицы на блок векторов. Поэтому распараллеливание должно быть прежде всего применено к этой операции.

Можно рассматривать два подхода к распараллеливанию умножения матрицы на блок векторов, связанных с распределенным хранением самой матрицы и (или) распределенным хранением блока.

Необходимость использования первого из отмеченных подходов продиктована также невозможностью хранения в оперативной памяти одного вычислительного узла всей исследуемой матрицы. Для использования второго подхода рассмотрим блок из s векторов, а именно — будем считать, что во всех рассматриваемых алгоритмах решается система линейных уравнений вида $DX = 0$, где $D \in \mathbb{F}^{N \times N}, X \in \mathbb{F}^{N \times s}$.

Рассмотрим задачу параллельного распределенного умножения разреженной матрицы на блочный вектор.

Для обеспечения баланса нагрузки необходимо делить разреженную матрицу на части, содержащие приблизительно одинаковое число единиц. Таким образом, без дополнительных преобразований матрицу можно разрезать только в одном направлении, по соотношениям. Разделим матрицу $D \in \mathbb{F}^{N \times N}$ рассматриваемой системы ($A = D^T D$) линейных однородных уравнений на горизонтальные полоски по числу l используемых вычислительных узлов. Таким образом, i -му процессору достанется матрица D_i , в которой ненулевыми оставлены лишь соответствующие N/l строк матрицы D .

На первом этапе нового алгоритма требуется вычислить $(D^T D)^i B, B^T (D^T D)^i B, B \in \mathbb{F}^{N \times s}, i = 1, 2, \dots$. Тогда

$$D^T D = \sum_{i=1}^l D_i^T D_i; D^T D B_j = \sum_{i=1}^l D_i^T D_i B_j, j = 1, \dots, s, \quad (9)$$

где $B_j \in \mathbb{F}^N$ — это j -я вертикальная полоска блока $B \in \mathbb{F}^{N \times s}$. Для хранения этой полоски и матрицы D_i на соответствующем вычислительном узле потребуется память $Nn + \frac{nNd}{l}$ бит (считаем, что для каждого ненулевого элемента матрицы D_i хранится его номер строки, номер столбца и содержание в одном "большом" машинном слове длины n).

Таким образом, распределенное вычисление блочного вектора $D^\top DB$ на ls вычислительных узлах потребует $\frac{N}{l}d$ операций в \mathbb{F} для вычисления элементов $D_i B_j$ и еще $\frac{Nd}{l}$, по числу ненулевых элементов

в матрице D_i^\top , для завершения вычисления $D_i^\top D_i B_j \in \mathbb{F}^N$. Отметим здесь, что для первого умножения (на D_i) при такой схеме расчетов не нужны пересылки, поэтому оно занимает существенно меньше времени, чем второе, требующее получения вектора AB_j . Для этого требуется пересылка со сложением по формуле (9) и обратная рассылка полученной левой части этого равенства на все l задействованных в ее вычислении вычислительных узла. Используя алгоритм циклической пересылки [6] для вычисления сложения в формуле (9), получим общий объем последовательных пересылок машинных слов со сложением не более $\frac{c_1}{n} 2Nn = 2Nc_1$. Действительно,

в соответствии с алгоритмом циклической пересылки, для получения итоговой суммы на каждом вычислительном узле группы требуется, чтобы этот узел дважды получил и передал вектор, занимающий память с размерами не более чем $N \times 1$ по двустороннему симметричному каналу связи.

Суммируя, получим итоговую оценку времени на вычисление $(D^\top D)B$ на ls вычислительных узлах:

$$\frac{2Nd}{l} + 2Nc_1. \quad (10)$$

В предлагаемой версии распределенного вычисления векторов $(D^\top D)B$ на каждом из l вычислительных узлов j -й группы, $1 \leq j \leq s$, в некоторый момент содержится блочный вектор $(D^\top D)^i B_j$, $i \in \{1, \dots, k\}$. Совокупность этих векторов будем называть куском пространства Крылова. При вычислении этого куска, в случае нового алгоритма, все их можно хранить распределенно на вычислительных узлах группы с номером j по $\frac{N}{l}$ строк на каждом.

Поскольку все векторы $(D^\top D)^i B_j$, $i \in \{1, \dots, k\}$ в ходе вычислений появляются на каждом узле группы, часть их строк для этого можно просто забыть.

Для распределенного по строкам (полоскам) хранения на l вычислительных узлах всего блока B^\top потребуется еще $\frac{Nns}{l}$ бит. После вычисления вектор-столбца $(D^\top D)^i B_j$ на каждом из l вычислительных

узлов j -й группы вычисляется своя часть скалярного произведения $B^\top (D^\top D)^i B_j$ не более, чем за $\frac{Ns}{l}$ операций в \mathbb{F} .

Все результаты вычисления j -й группы после конкатенации на некотором вычислительном узле правой части вычислительной системы, содержащем k старших коэффициентов $Q_j^{(i)}$, дают $B^\top (D^\top D)^i B_j$, которые и будем использовать при решении уравнений (2) относительно столбцов переменных μ_i, γ_j на правой части вычислительной системы (т. е. $\geq s$). Предварительно на t вычислительных узлах правой части вычислительной системы параллельно обращаются матрицы $Q_k^{(k)}, Q_{k-1}^{(k-1)}$, для подстановки их в систему (2), а затем и матрицы $W_i^\top A W_i = Q_i^{(i)\top} P_{t+1}$, $i = 1, \dots, k$, для дальнейшей подстановки их в формулу для решения X . Пересылки при этом незначительные ввиду маленьких размеров пересылаемых матриц. Добавляя к оценке (10), получим оценку времени на вычисление очередного блока векторов из пространства Крылова и попарных скалярных произведений на кластере из sl вычислительных узлов:

$$\frac{2dN}{l} + \frac{Ns}{l} + 2Nc_1.$$

Если

$$\max \left\{ \frac{d}{c_1}, \frac{s}{2c_1} \right\} < l, \quad (11)$$

то первое и второе слагаемые этой оценки меньше третьего. Учитывая, что общее число шагов всего алгоритма будет $2 \frac{N}{s}$, получаем общую оценку времени работы первого этапа (вычисление пространства Крылова и скалярных произведений):

$$\frac{8N^2 c_1}{s}. \quad (12)$$

Необходимая память на каждом из sl вычислительных узлов левой части вычислительной системы (с учетом распределенного хранения куска пространства Крылова и исходной матрицы D) определяется размерами соответствующих матриц и форматом хранения их элементов, которые были введены выше:

$$\begin{array}{cccc} \frac{nNd}{l} & Nn & \frac{Nns}{l} & \frac{Nnk}{l} \\ \text{Матрица } D_j & \text{Текущее } B_j & \text{Часть } B^\top & \text{Кусок пр. Крылова} \\ & & = & \\ & & = \frac{Nn}{8 \cdot 10^9} \left(1 + \frac{d+s+k}{l} \right) \text{ ГБ.} & \end{array}$$

В случае $N \approx 2^{24}$ это составляет примерно $\frac{n}{128} \left(1 + \frac{d+s+k}{l} \right)$ ГБ.

После того как на правой части вычислительной системы будут получены все необходимые коэффициенты, на левой части вычислительной системы вычисляется часть решения по формуле (7) и новые образующие по формуле (8). Пересылки при этом происходят параллель-

но по медленным каналам. Для загрузки левой части вычислительной системы, занимающейся умножением матрицы на блочный вектор и вычислением скалярных произведений, потребуется всего не более чем

$$c_2 \cdot 3 \frac{N}{l} s \frac{N}{ks} = 3 \frac{N^2 c_2}{kl} \quad (13)$$

пересылок на передаче (рассылка частей (по вертикали) векторов $Q^{(k-1)}(A, B)$, $Q^{(k)}(A, B)$ по медленным горизонтальным связям и получение решения по формуле (7)). Сам параллельный пересчет двух образующих и отрезка решения по формулам (7) и (8) с помощью умножения на A_1 , т. е. умножения матрицы из $\mathbb{F}^{\frac{N}{l} \times k}$ на матрицу из $\mathbb{F}^{k \times s}$ займет $3 \frac{Nks}{l}$, а по всему алгоритму $3 \frac{N^2}{l}$, что выбором l может быть сделано достаточно маленьким, хотя и не зависящим от k . Поскольку параметр l не увеличивает время ни в какой операции рассматриваемого алгоритма, то его можно наращивать, пока имеются свободные вычислительные узлы. Как следствие, теоретически время на выполнение этого последнего в цикле шага можно уменьшить до нуля без ущерба для остальных шагов.

Работа правой части вычислительной системы

Теорема 1. Оценка времени работы параллельной реализации с использованием блочного фактора нового алгоритма при достаточно большом N и достаточно большом числе вычислительных узлов имеет вид

$$48c_1 N^{3/2}$$

Доказательство. Для правой части вычислительной системы возьмем $t = ks$. Для удобства будем рассматривать k групп по s узлов. Пусть на каждой группе вычислительных узлов правой части вычислительной системы при вычислении $(t+1)$ -го блока пространства Крылова хранятся коэффициенты $Q_{k,i}^{(T)}, Q_{k-1,i}^{(T)}, Q_i^{(T)}$ для $T = t, t-1$, и некоторого фиксированного значения i , а также все μ и γ по мере их вычисления. Поскольку μ и γ используются последовательно, их можно не хранить в одном месте, а получив и использовав, переслать другим группам вычислительных узлов. Для того чтобы хранить все указанные коэффициенты потребуется на ks вычислительных узлах $O(ns^2)$ свободных бит оперативной памяти на каждой группе из s вычислительных узлов. На втором этапе вычислительные узлы правой части вычислительной системы будут заниматься вычислениями по формулам (2), распределено по столбцам, и по формуле (3), вычисляя только старший коэффициент $Q_i^{(t)}$ многочлена на каждом шаге и k старших коэффициентов для последних двух многочленов в цикле между синхронизациями. Вычислениями по формулам (4) и нахождением v_i по формуле (5) занимается еще один выделенный узел правой части вычислительной системы, который также вычисляет коэффициен-

ты при элементах с крышками $((W_t^T AW)_t)^{-1} W_t^T B$ (7)). При этом обращение и произведение матриц размера $s \times s$ можно проводить параллельно на всех t узлах правой части вычислительной системы. Пересылки при этом незначительные и учитывать их не будем.

Каждая группа с номером i_0 из s узлов правой части вычислительной системы выполняет перечисленные далее операции.

1. На выделенной группе узлов с использованием s узлов правой части вычислительной системы вычисляется вспомогательный коэффициент $(W_{t+1}^T AW_{t+1})^{-1} W_{t+1}^T B$ с помощью обращения матрицы $W_{t+1}^T AW_{t+1} = Q_{t+1}^{(t+1)\top} \sigma_{t+2}$ и рекуррентных формул для $W_t^T B$ (см. формальное описание алгоритма). Итого три умножения и одно обращение за время, не превышающее $10,5s^2c_1$.

2. Получает по бинарному дереву коэффициенты рекуррентных формул (два коэффициента v_0, v_{-1}) и вспомогательный коэффициент $(W_{t+1}^T AW_{t+1})^{-1} W_{t+1}^T B$.

Пересчитываем свои (при фиксированном i_0) коэффициенты $Q_{k,i_0}^{(t+1)}, Q_{k-1,i_0}^{(t+1)}$ по рекуррентным формулам (5) (для этого один дополнительный коэффициент получает с соседнего узла, учитывая сдвиг вверх крайнего левого столбца в формуле (5)), и затем свою координату вектора \hat{X} (см. (7)). Всего не более шести умножений. Произведение матриц $s \times s$ один узел может проводить с использованием s узлов своей группы правой части вычислительной системы за время, не превышающее $3s^2c_1$ (передать элементы множителей, получить элементы результата).

3. Вычисляет сумму слагаемых в формулах вида (4):

$$\sum_{i=0}^{t+1-k} \mu_{i+j} Q_{k,i}^{(t+1)} + \sum_{i=0}^{t-k} \gamma_{i+j} Q_{k-1,i}^{(t+1)}, \quad j = t+2, t+3,$$

содержащих коэффициенты $Q_{k,i_0}^{(t+1)}, Q_{k-1,i_0}^{(t+1)}$, которые хранятся на данном узле (четыре умножения).

4. Передает эту сумму на выделенную группу процессоров в правой части вычислительной системы для решения очередных систем линейных однородных уравнений в верхних частях матричных равенств вида (5), а выделенная группа решает указанную систему с использованием всех $t = ks$ узлов правой части вычислительной системы за время, не превышающее $3/2$ от $(3s)^2 c_1$ (привести к треугольному виду и решить на s вычислительных узлах). Здесь подразумевается, что t больше, чем $3s$ (размер системы), так что время на решение с помощью параллельного алгоритма меньше указанного значения. Полученное рассыпается обратно на k групп узлов по бинарному дереву. Поскольку эти коэффициенты небольшие, их пересылками пренебрежем.

Объем необходимой памяти на одной группе из s узлов правой части вычислительной системы оценивается значением величины $3kns^2$. Здесь учтено по два вычисляемых коэффициента разложений для приближений $Q^{(T)}$, для $T = t, t-1$, один вспомогательный коэффициент, а также все μ и ν (k матриц размера $s \times s$), а также резерв для пересчета. Все μ и ν можно не хранить, а получать, использовать (умножать на соответствующие коэффициенты $Q_i^{(T)}$, хранящиеся на i -м вычислительном узле) и пересыпать дальше, получая новые, в этом случае память оценивается как $O(ns^2)$.

Соответствующие оценки времени вычислений по перечисленным пунктам и операциям следующие:

- 1) $10,5Nsc_1$;
- 2) $18s^2c_1 \frac{N}{s} = 18Nsc_1$;
- 3) $12s^2c_1 \frac{N}{s} = 12Nsc_1$;
- 4) $\frac{27}{2}s^2c_1 \frac{N}{s} = \frac{27}{2}Nsc_1$.

Складывая все значения, получим следующую оценку времени работы:

$$54Nsc_1. \quad (14)$$

Перед очередной синхронизацией осуществляется пересылка. Поскольку решение и образующие хранятся частями вида $F^{\frac{N}{l}}$, то получаемые для пересчета коэффициенты должны быть в F^{kxs} . Пересылка здесь незначительная.

В сумме с (14) и (13) время вычислений не более

$$54Nsc_1 + 3 \frac{N^2c_2}{kl}. \quad (15)$$

Здесь же учтем вычисления μ_j, γ_j по формулам (2) (обращение двух матриц размера $s \times s$ и решение блочнотреугольной линейной системы уже без обращений) при известных левых частях этих формул на правой части вычислительной системы с относительно быстрыми связями по всему алгоритму $\left(2\frac{3}{2}s^2c_1 + 3s^2c_1k2\right)\frac{N}{sk} < 9Nsc_1$. Здесь принят во внимание тот факт, что каждый коэффициент $Q_{i_0}^{(T)}$ используется в системе (2) не более k раз.

Вычисление и обмен, в результате которого узлы, вычисляющие скалярные произведения, полностью получат старшие k коэффициентов многочленов $Q^{(k)}(\lambda)$, аналогично займет не более $(3s^2c_1k)\frac{N}{sk} = 3Nsc_1$.

Общая оценка с учетом выражений (12), (15) — $12Nsc_1 + 54Nsc_1 + 3 \frac{N^2c_2}{kl} + \frac{8N^2c_1}{s}$.

Для минимизации получившейся функции по k возьмем $k > 3c_2$, получим общую оценку времени:

$$66Nsc_1 + \frac{8N^2c_1}{s}. \quad (16)$$

Естественным ограничением сверху на k является l — чтобы число вычислительных узлов на правой части вычислительной системы не стало слишком большим. Это обстоятельство позволяет ограничить память узлов левой части вычислительной системы, на которой хранится кусок пространства Крылова, до реальных размеров (см. выше).

Выбрав

$$s = \sqrt{\frac{8N}{66}}, \quad (17)$$

получим $48c_1N^{3/2}$,

Теорема доказана.

В процессе доказательства суммировалось время по двум этапам, хотя для начала второго этапа достаточно первых элементов рассматриваемого отрезка пространства Крылова, что следует из равенств (2), (3), (4). С учетом этого факта первый и второй этапы можно проводить практически параллельно. При согласованной работе узлов, занятых на первом и втором этапах, общее время можно уменьшить примерно вдвое.

Число арифметических операций на втором этапе алгоритма Видемана—Копперсмита—Томе [7] оценено в работе [8] значением

$$O(Nns(ns + \log_2 N) \log_2 N \log_2 \log_2 N).$$

Складывая с (12) и оптимизируя по s , получаем для всего алгоритма оценку

$$O\left(N^{1+\frac{2}{3}} (\log_2 N \log_2 \log_2 N)^{\frac{1}{3}}\right).$$

Кроме того, работа программы при постановке рекорда целочисленной факторизации в 2012 г. потребовала использования критического объема оперативной памяти, около 1 ТБ. Это связано с ростом коэффициентов многочленов большой степени при использовании алгоритма быстрого преобразования Фурье для их умножения. Совсем недавно, используя высокопроизводительную технику, группой под руководством Томе были поставлены рекорды целой факторизации (RSA-250) и дискретного логарифмирования по модулю простого числа в 240 десятичных знаков (подробных публикаций пока нет). В рассматриваемом алгоритме произведение многочленов есть, но их степени меньше в число синхронизаций раз. Таким образом, предложенный алгоритм лучше алгоритма Видемана—Копперсмита асимптотически по времени работы и по используемой оперативной памяти.

Здесь следует сделать следующее замечание. Оптимальное число вычислительных узлов получается приблизительно равным $\frac{N}{33c_1}$ (см. (11)), что при актуальных значениях параметров приблизительно равно 13 000 узлам. Поэтому совсем не обязательно выбирать s оптимальным согласно формуле (17). Если s выбрано меньше, то оценка времени работы алгоритма (см. (16)) будет $\frac{8N^2c_1}{s}$ при $K = sl + t \approx \frac{s^2}{2c_1}$ (см. (16) при $k = 3c_2$, $t \approx 3c_2s$).

Заключение

Были получены оптимальные параметры настройки вычислительной системы для успешной работы программы решения большой разреженной системы над большим простым полем. Кроме того, получены оценки времени работы такой программы, лучшие из известных на сегодняшний момент.

Список литературы

1. Черепнёв М. А. Блочный алгоритм типа Ланцоша решения разреженных систем линейных уравнений // Дискретная математика. — 2008. — Т. 20, № 1. — С. 145–150.
2. Черепнёв М. А., Замарашкин Н. Л. Универсальный блочный метод Ланцоша—Паде для систем линейных уравнений над

большими простыми полями // Фундаментальная и прикладная математика. — 2014. — Т. 19, Вып. 6. — С. 223–247.

3. Montgomery P. L. A Block Lanczos Algorithm for Finding Dependencies over GF(2) // Advances in Cryptology — EuroCrypt'95 / Eds by L. C. Guillou, J.-J. Quisquater, Berlin: Springer-Verlag, Lect. Notes in Comp. Sci. 1995. — Vol. 921. — P. 106–120.

4. Cherepnev M. A. Some estimations of performance of parallel algorithms for solving large linear systems over GF(2) // A Journal of Tambov State University, The works of participants of International conference "ParCA" presented according to the results of reviewing by International Program Committee. — 2010. — Vol. 15, Iss. 4. — P. 134–135.

5. Черепнёв М. А. О некоторых вычислениях в пространствах Крылова над GF(2) // Вестник Тамбовского университета Сер. Естественные и технические науки. — 2009. — Т. 14. Вып. 4. — С. 833–835.

6. Barnett M., Littlefield R., Payne D. G., van de Geijn R. A. Global combine algorithms on mesh architectures with wormhole routing // Int. Par. Processing Symp. 1993-Apr. — P. 156–162.

7. Coppersmith D. Solving homogeneous linear systems over GF(2) via block Widemann algorithm // Mathematics of Computation. — 1994. — Vol. 62, No. 205. — P. 333–350.

8. Thome E. Subquadratic computation of vector generating polynomials and improvement of the block Wiedemann algorithm // Journal of Symbolic Computation. — 2002. — Vol. 33, No. 5. — P. 757–775.

Estimation of the Speed of a New Parallel Block Algorithm for Finding Solutions of the Large Sparse Linear Systems over a Large Prime Field

M. A. Cherepnev, Ph. D., Associate Professor, cherepniov@gmail.com,
Lomonosov Moscow State University, Moscow, 119992, Russian Federation

Corresponding author:

Cherepnev Michail A., Ph. D., Associate Professor, Lomonosov Moscow State University, Moscow, 119992, Russian Federation
E-mail: cherepniov@gmail.com

Received on June 17, 2020

Accepted on June 22, 2020

The question of how to estimate the running time of a program that implements an algorithm for solving a large sparse system of equations over a large prime field, described in [1, 2], on a computer system of K independent computing nodes is considered. The article considers the topology that connects nodes of the computer network, "torus". At the same time, we believe that in each of the two directions of this torus, there are connections with different bandwidth. In reality, this may correspond to large computing clusters connected by a relatively slow Internet network, or multiple multi-core (possibly with virtual cores) processors in the same cluster. At the same time, the estimates of working time take into account not only the time for the invoice, as in [2], but also the time for forwarding. The block nature of the algorithm allows us to reduce the running time of the sequential scalar version (Montgomery's algorithm [3]) by approximately \sqrt{K} times, with K growing to the size of the original problem. At the same time, about half of all switching is allowed to be slow, which makes it possible to dial the necessary number of computing nodes from different clusters. A specific distributed storage of intermediate results of calculations is proposed. This results in saving not only computing resources, but also time for transmission, which is carried out in parallel over independent branches of our network. In addition, the solution of dense systems of linear equations, which must be done at each step of our algorithm, as well as the inversion of dense matrices, we suggest using parallel calculations. This will reduce the conversion time of matrices of size $s \times s$ to $O(s^2)$ and the total running time of the entire program for solving

a sparse system of linear equations in a space of dimension N to $O\left(\frac{N^2}{\sqrt{K}}\right)$ for an arbitrary K , from 1 to N . Thus, theoretically, the optimal time $O(N^{3/2})$ is obtained.

Keywords: discrete logarithm method, Lanczos method, parallel computing, block algorithms

Acknowledgements: This work was supported by the Russian Foundation for Basic Research, project nos. 18-29-03421 мк.

For citation:

Cherepnev M. A. Estimation of the Speed of a New Parallel Block Algorithm for Finding Solutions of the Large Sparse Linear Systems over a Large Prime Field, *Programmnaya Ingeneriya*, 2020, vol. 11, no. 4, pp. 242–250

DOI: 10.17587/prin.11.242-250

References

1. Cherepnev M. A. Block algorithm of lanczos type for solving sparse systems of linear equations, *Diskretnaya matematika*, 2008, vol. 20, no. 1, pp. 145–150 (in Russian).

2. Cherepnev M. A., Zamarashkin N. L. Universal block method of lanczos-Pade for systems of linear equations over large simple fields, *Journal of Mathematical Sciences*, 2017, vol. 221, no. 3, pp. 461–478.

3. Montgomery P. L. A Block Lanczos Algorithm for Finding Dependencies over GF(2), *Advances in Cryptology — EuroCrypt'95* / Eds by L. C. Guillou, J.-J. Quisquater, Berlin, Springer-Verlag, Lect. Notes in Comp. Sci., 1995, vol. 921, pp. 106–120.

4. Cherepnev M. A. Some estimations of performance of parallel algorithms for solving large linear systems over GF(2), *A Journal of Tambov State University, The works of participants of International conference "ParCA" presented according to the results of reviewing by International Program Committee*, 2010, vol. 15, iss. 4, pp. 1342–1353.

international conference "ParCA" presented according to the results of reviewing by International Program Committee, 2010, vol. 15, iss. 4, pp. 1342–1353.

5. Cherepnev M. A. Some calculations in Krylov space over GF(2). *Vestnik Tambovskogo universiteta Ser. Estestvennye i tekhnicheskie nauki*, 2009, vol. 14, no. 4, pp. 833–835 (in Russian).

6. Barnett M., Littlefield R., Payne D. G., van de Geijn R. A. Global combine algorithms on mesh architectures with wormhole routing, *Int. Par. Processing Symp.*, 1993-Apr., pp. 156–162.

7. Coppersmith D. Solving homogeneous linear systems over GF(2) via block Widemann algorithm, *Mathematics of Computation*, 1994, vol. 62, no. 205, pp. 333–350.

8. Thome E. Subquadratic computation of vector generating polynomials and improvement of the block Wiedemann algorithm, *Journal of Symbolic Computation*, 2002, vol. 33, no. 5, pp. 757–775.

С. И. Носков, д-р техн. наук, проф., sergey.noskov.57@mail.ru, **А. С. Вергасов**, ассистент, tluck@inbox.ru, Иркутский государственный университет путей сообщения

Регрессионная модель структурных факторов киберугроз

Исследования, результаты которых представлены в статье, основаны на статистическом анализе данных по киберугрозам, классифицированным по трем группам факторов: мотивы злоумышленников, категории жертв, объекты атак. Такая классификация принята в компании Positive Technologies, которая специализируется в области разработки систем информационной безопасности и комплексного анализа современных тенденций в реализации киберугроз. В результате обработки поквартальной статистической информации (в период с I квартала 2017 г. по II квартал 2019 г.) построена открытая регрессионная рекурсивная модель динамики 19 структурных факторов киберугроз, позволяющая решать широкий круг проблем анализа и прогнозирования. Представленные результаты являются первым этапом исследований на этом направлении. В дальнейшем авторы намереваются их продолжить в целях выработки конкретных рекомендаций для предотвращения или отражения кибератак.

Ключевые слова: регрессионная модель, киберугрозы, Cyber Kill Chain, критерии адекватности, прогнозирование, рекурсия

Введение

В современную цифровую эпоху население и служащие частных, а также государственных учреждений развитых стран повседневно используют цифровые устройства — телефоны, компьютеры, банкоматы, которые могут хранить в себе персональные, учетные данные и другие виды ценной информации. К их числу относятся данные банковских карт, пароли от учетных записей, частные фото и видеоматериалы, информация о состоянии здоровья и т. д. Эта информация потенциально может быть подвержена кибератакам со стороны злоумышленников.

По своей структуре кибератаки могут значительно отличаться друг от друга. Еще в 2000 г. ущерб от вируса ILoveYou составил около 10 млрд долл. США [1]. В 2016 г. DDoS-атаки были зафиксированы в каждом четвертом банке (26 %). Среди других финансовых структур вредному воздействию подверглось 22 % компаний. При этом усредненный ущерб для кредитных организаций составил 1 172 000 долл. США в расчете на один банк [2], а средняя стоимость убытков при взломе крупной компании — 6,3 млн долл. США [3]. К концу 2016 г. киберпреступность обходилась мировой экономике в 650 млрд долл. США [4].

По информации Сбербанка России в 2019 г. ущерб компаний от кибератак увеличился в 1,6 раза по сравнению с 2018 г. В 2018 г. убытки компаний достигли 1,5 трлн долл. США, в этом году они могут составить 2,5 трлн долл. США [5].

Эти обстоятельства предопределяют высокую актуальность исследований, направленных на изучение современных тенденций в динамике киберпреступности. Настоящая работа направлена на их выявление методами математического моделирования.

При подготовке публикации были изучены работы, описывающие существующие модели злоумышленни-

ков [19, 20], модель жизненного цикла кибератак Cyber Kill Chain [18], различные ее модификации [23, 24], основанное на таких моделях прогнозирование кибератак и защиту от них на железнодорожных информационных ресурсах [21]; а также структуру защиты от кибератак в компьютерных сетях [22].

Описание факторного пространства

В качестве информационной базы модели использовались статистические поквартальные данные компании Positive Technologies, занимающейся консалтингом и разработкой программного обеспечения в области информационной безопасности, за период с I квартала 2017 г. по II квартал 2019 г., публикуемые в открытом доступе [5]. Полный объем обрабатываемой информации составил десять наблюдений. При этом были проанализированы данные по крупным клиентам (в том числе по юридическим лицам) компании, которые подвергались до 150 атакам в месяц.

Всего выделено три группы факторов: мотивы злоумышленников $X_1 = \{x_1, \dots, x_4\}$; категории объектов, подвергающихся атакам $X_2 = \{x_5, \dots, x_{13}\}$, собственно объекты атак $X_3 = \{x_{14}, \dots, x_{19}\}$.

Рассмотрим эти факторы более подробно.

1. Мотивы злоумышленников в группе X_1 представлены на рис. 1, см. третью сторону обложки.

К мотивам x_1 относятся: получение данных, в том числе персональных, личной переписки и другой информации с последующим извлечением прибыли, например, с помощью шантажа (рис. 2).

Подгруппа мотивов x_2 — финансовая выгода, как правило, непосредственное списывание средств со счетов (рис. 3).

Мотивы x_3 — хактивизм, здесь целями атак становятся политические позиции физических и юридиче-

ских лиц, взгляды и действия которых не устраивают киберпреступников (например, атаки на американские и европейские государственные учреждения после запуска проекта WikiLeaks) (рис. 4).

К мотивам x_4 относятся проявления элементов кибервойны. Целью таких атак является компрометация и дестабилизация работы конкурентов, хищение коммерческой тайны (рис. 5).

2. В группе X_2 выделяют перечисленные далее подгруппы объектов, подвергшихся атакам.

Частные лица, пострадавшие от кибератак, формируют подгруппу x_5 . Главной целью злоумышленника при атаке является получение учетных и персональных данных.

К объектам атак x_6 относятся государственные учреждения, которые чаще всего являются объектом атак при кибершпионаже или интернет-вымогательстве. Посредством атак злоумышленники получают доступ к коммерческим тайнам, персональным и учетным данным сотрудников.

Действия, направленные на получение информации о клиентах, счетах, транзакциях, доступе к их выполнению и иные нелегитимные операции в финансовой отрасли входят в подгруппу x_7 .

Атаки на медицинские учреждения образуют подгруппу x_8 . Целью атак, как правило, становится получение персональных данных, учетных данных и конфиденциальной информации специального характера.

Объекты сферы образования, составляющие подгруппу x_9 , атакуются ради получения учетных и персональных данных.

В подгруппе x_{10} объектами, подвергшимися атакам, являются онлайн-сервисы. Характерная цель этих атак — получение персональных данных и данных платежных карт.

Пострадавшие предприятия сферы услуг относятся к подгруппе x_{11} . Здесь преобладают атаки в целях шантажа и вымогательства.

Промышленные компании, подвергающиеся атакам ради учетных данных и коммерческих тайн, DDoS-атакам в целях вымогательства и иным видам нападений, формируют подгруппу x_{12} .

Кибератаки без привязки к конкретной отрасли, в том числе гибридные масштабные атаки, затрагивающие несколько отраслей, относятся к подгруппе x_{13} .

3. В группе факторов X_3 кибератаки распределяются по характеризующему их объекту атаки.

Так, к подгруппе x_{14} (инфраструктура) относятся: компьютеры, серверы и сетевое оборудование.

Веб-ресурсы, а именно сайты и сервисы составляют подгруппу x_{15} .

Пользователи составляют подгруппу x_{16} , телефоны, планшеты и другие мобильные устройства — подгруппу x_{17} .

Подгруппа x_{18} — это IoT-техника, подключенная к сети (например, камеры видеонаблюдения).

Наконец, в подгруппе x_{19} объектами являются банкоматы и POS-терминалы.

Таким образом, вся подлежащая обработке информация представляет собой матрицу \mathbf{X} вида



Рис. 2. Доля фактора x_1 в группе X_1



Рис. 3. Доля фактора x_2 в группе X_1

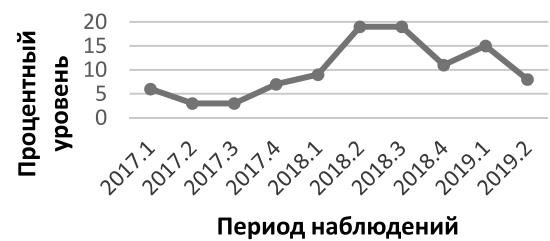


Рис. 4. Доля фактора x_3 в группе X_1



Рис. 5. Доля фактора x_4 в группе X_1

$$\mathbf{X} = \|x_{ik}\|, k = \overline{1, 10}, i = \overline{1, 19}$$

Все значения x_{ik} выражены в процентах, причем внутри каждой группы их сумма составляет 100 %:

$$\sum_{i=1}^4 x_{ik} = 100, \sum_{i=5}^{13} x_{ik} = 100, \sum_{i=14}^{19} x_{ik} = 100, k = \overline{1, 10}.$$

Таким образом, все перечисленные выше факторы имеют не абсолютный, а структурный характер.

Математическая модель

Прежде чем приступить собственно к описанию модели, введем некоторые обозначения. Наряду с определенными ранее переменными x_1, \dots, x_{19} введем новую переменную x_{20} по правилу

$$x_{20k} = k + 1, k = \overline{1, 10}. \quad (1)$$

В физическом смысле x_{20} — фактор времени. Именно такой характер его формирования вызван использованием при моделировании элементарной операции \ln . Сам же этот фактор весьма значим при моделировании динамических процессов.

При оценке адекватности регрессионных уравнений используются традиционные (см., например, работы [6–11]) для анализа данных критерии множественной детерминации R^2 , Фишера F и Стьюдента t . При этом выражение $\mathbf{T} = (a; b; c)$ означает, что для соответствующей трехфакторной регрессии (или двухфакторной регрессии со свободным членом) значение t -критерия для первого параметра равно a , для второго — b , а для третьего — c .

При построении модели был применен программный комплекс GRETl [12], а также методы и алгоритмы, описанные в работах [13–17]. Вид аппроксимирующей функции для каждой экзогенной переменной подбирался с помощью программы, реализующей модификацию алгоритма, представленного в работе [16]. При этом наряду с исходными переменными широко используются их различные элементарные преобразования.

Построение модели основывается на следующем смысловом посыле — изначально злоумышленником движут его мотивы, а именно, в чем вообще состоит цель организуемой им атаки. В зависимости от этого он выбирает категорию жертвы. И лишь после этого назначается объект атаки, т. е. посредством каких, в частности, технических средств она будет осуществлена. Именно таким образом определяется последовательность вхождения в модель всех перечисленных групп факторов.

Математическая модель регрессионного типа структурных факторов киберугроз, построенная с помощью упомянутой выше программы, имеет следующий вид:

$$x_1 = 72,544 + 4490 \left(\frac{1}{x_3 x_{20}^3} \right) - 832 \left(\frac{\ln(x_{20})}{x_{20}^2} \right), \quad (2)$$

$$\mathbf{T} = (21,09; 7,09; -9,317), R^2 = 0,955, F = 74,$$

$$x_2 = 88,743 - 0,216(x_1 \ln(x_1)) - 0321(x_3 \ln(x_3)), \\ \mathbf{T} = (106,1; -33,14; -15,46), R^2 = 0,997, F = 1310, \quad (3)$$

$$x_4 = 100 - x_1 - x_2 - x_3, \quad (4)$$

$$x_5 = -26,051 - 7802 \left(\frac{1}{x_2 x_{20}^2} \right) + 14,065 \left(\ln(x_2) + \frac{1}{x_3} \right), \quad (5)$$

$$\mathbf{T} = (-5,94; -15,4; 12,23), R^2 = 0,973, F = 126,$$

$$x_6 = -2,691 - 43,011 \left(\frac{1}{x_3^2} + \frac{1}{x_4^2} \right) + 58,741 \left(\frac{1}{x_4} + \frac{1}{x_{20}^3} \right), \quad (6)$$

$$\mathbf{T} = (-0,97; -7,42; 6,79), R^2 = 0,918, F = 39,$$

$$x_7 = -0,29 - 0,00265 \left(\frac{x_{20}^3}{x_4} \right) + 7,894 \left(\ln(x_4) + \frac{1}{x_4^2} \right), \quad (7)$$

$$\mathbf{T} = (-0,18; -5,07; 4,94), R^2 = 0,904, F = 33,$$

$$x_8 = 100 - x_5 - x_6 - x_7 - x_9 - x_{10} - x_{11} - x_{12} - x_{13}, \quad (8)$$

$$x_9 = -0,485 + 0,00316 \left(\frac{x_1^3}{x_2^2} \right) + 380 \left(\frac{x_3^2}{x_1^3} \right), \quad (9)$$

$$\mathbf{T} = (1,06; 13,64; 6,42), R^2 = 0,963, F = 93,$$

$$x_{10} = 33 + 1014 \left(\frac{x_{20}}{x_2^2} \right) - 5,15 (\ln(x_1) + \ln(x_{20})), \quad (10)$$

$$\mathbf{T} = (12,8; 7,7; 10,6), R^2 = 0,949, F = 65,$$

$$x_{11} = -4,719 + 163715 \left(\frac{1}{x_2^3 x_4} \right) + 0,12 (x_2 + x_4^3), \quad (11)$$

$$\mathbf{T} = (7,62; 19,7; 13,51), R^2 = 0,967, F = 105,$$

$$x_{12} = -50,47 + 0,00406 (x_1^3 \ln(x_3)) + \\ + 12,129 \left(\ln(x_2) + \frac{1}{x_{20}^2} \right), \quad (12)$$

$$\mathbf{T} = (-8,8; 11,9; 9,27), R^2 = 0,964, F = 94,$$

$$x_{13} = 51,33 + 333 \left(\frac{1}{x_4^3 x_{20}^3} \right) - 0,00524 (x_1^2 + x_2^2), \quad (13)$$

$$\mathbf{T} = (24,35; 7,9; 10,8), R^2 = 0,947, F = 63,$$

$$x_{14} = 63,296 - 8,784 \left(\frac{x_{10}}{x_{12}} \right) - 33,519 \left(\frac{\ln(x_4)}{x_4^2} \right), \quad (14)$$

$$\mathbf{T} = (941,75; -12,9; -4,77), R^2 = 0,959, F = 83,$$

$$x_{15} = 100 - x_{14} - x_{16} - x_{17} - x_{18} - x_{19}, \quad (15)$$

$$x_{16} = 20,32 - 869141 \left(\frac{1}{x_{11}^2 x_{13}^3} \right) - 99,352 \left(\frac{x_1}{x_2^2} \right), \quad (16)$$

$$\mathbf{T} = (75,2; -15,1; 16,8), R^2 = 0,986, F = 261,$$

$$x_{17} = 11,174 - 0,0116 \left(\frac{x_6^3}{x_5} \right) - 0,000488 (x_1^2 + x_3^3), \quad (17)$$

$$\mathbf{T} = (57,5; -22,7; -14,55), R^2 = 0,988, F = 299,$$

$$x_{18} = -10,169 + 0,0000629 \left(\ln(x_8) x_{13}^3 \right) + \\ + 2,475 \left(\ln(x_2) + \frac{1}{x_3} \right), \quad (18)$$

$\mathbf{T} = (-6,062; 9,945; 7,114), R^2 = 0,934,$
 $F = 49,702,$

$$x_{19} = 3,085 - 15319,727 \left(\frac{1}{x_6^3 x_{11}} \right) + 127,192 \left(\frac{1}{x_3^3 x_4} \right), \quad (19)$$

$\mathbf{T} = (18,04; -9,7; 16,4), R^2 = 0,974, F = 136.$

Анализ значений критериев адекватности каждого уравнения модели (2)–(19) показывает, что они весьма высоки, что делает вполне правомерным ее практическое применение для решения проблем, связанных с анализом и прогнозированием.

Правые части регрессионных уравнений линейны по параметрам, но существенно нелинейны по независимым переменным. Попытки построения обычных линейных регрессий во всех случаях успехом не увенчались. Для примера приведем линейный аналог лишь одного из них — для переменной x_5 , причем даже с большим числом параметров:

$$x_5 = -41,7 + 0,74x_2 + 0,329x_3 + 3,73x_{20}, \quad (20)$$

$\mathbf{T} = (-1,09; 1,73; 0,51; 1,9), R^2 = 0,42, F = 1,47.$

Нетрудно убедиться в том, что уравнение (20) существенно уступает по значениям всех критериев адекватности своему нелинейному по переменным аналогу — регрессии (5).

Следует отметить одно важное обстоятельство. При построении каждого из уравнений (2) — (19) с использованием упомянутой выше программы перебираются миллионы их альтернативных вариантов с последующим выбором лучших из них, исходя из значений критериев адекватности. Уравнения (4), (8), (15) имеют не регрессионный, а балансовый характер, что следует из упомянутой выше структурности информационной базы модели.

Процесс построения модели (2) — (19) организован таким образом, чтобы обеспечить ее рекурсивность, которая давала бы возможность выстраивать последовательную схему расчетов значений эндогенных переменных при их прогнозировании без необходимости решения системы нелинейных уравнений. Порядок рекурсии при таких расчетах следующий:

- вход — x_3, x_{20} ;
- первый уровень — x_1 ;
- второй уровень — x_2, x_9 ;
- третий уровень — x_4, x_5, x_{10}, x_{12} ;
- четвертый уровень — $x_6, x_7, x_{11}, x_{13}, x_{14}$;
- пятый уровень — $x_8, x_{16}, x_{17}, x_{19}$;
- шестой уровень — x_{18} ;
- седьмой уровень — x_{15} .

В соответствии с этим порядком при прогнозировании вначале по некоторой заранее выбранной стратегии задаются значения входных переменных на прогнозном периоде, затем вычисляются значения переменных первого уровня, потом второго и т. д.

Заключение

Следует отметить исключительно важное, по мнению авторов, обстоятельство — приведенная в работе регрессионная модель киберугроз по степени своей адекватности близка к функциональной ввиду близости к единице значений критерия множественной детерминации для всех зависимостей при высоких же значениях критерия Фишера. Причем, что не менее важно, это достигнуто не за счет уменьшения числа их степеней свободы, что делать было категорически нельзя ввиду малой длины динамических рядов. Этот результат достигнут посредством организации с помощью соответствующей вычислительной технологии конкурса моделей из нескольких миллионов их альтернативных вариантов. Проведенная работа является необходимым начальным этапом для последующих исследований в области прогнозирования и разработки мер по эффективному отражению кибератак.

Список литературы

1. **Громкие** случаи кибератак и взломов в мире в 2000–2013 гг. URL: <https://ria.ru/20130809/955343983.html> (дата обращения 19.11.2019).
2. **Потери** организаций от киберпреступности. URL: http://www.tadviser.ru/index.php/Статья:Потери_организаций_от_киберпреступности (дата обращения 19.11.2019).
3. **Прошин А.** Взломать за 60 секунд: технологические тренды в обеспечении кибербезопасности // РБС. 24. 12. 2018. URL: <https://rb.ru/opinion/trendy-kiberbezopasnost/> (дата обращения 19.11.2019).
4. **К концу** 2016 года киберпреступность "обойдется" мировой экономике в \$650 миллиардов. URL: <https://www.croc.ru/news/detail/66901/> (дата обращения 19.11.2019).
5. **Потери** компаний от кибератак в мире в 2019 году могут достигнуть \$2,5 трлн. URL: <https://www.kommersant.ru/doc/3957187/> (дата обращения 19.11.2019).
6. **Дрейпер Н., Смит Г.** Прикладной регрессионный анализ. Множественная регрессия. 3-е изд. — М.: Диалектика, 2007. — 912 с.
7. **Ферстер Э., Ренц Б.** Методы корреляционного и регрессионного анализа. — М.: Финансы и статистика, 1981. — 302 с.
8. **Радченко С. Г.** Методология регрессионного анализа: — Киев: Корнийчук, 2011. — 376 с.
9. **Носков С. И.** Технология моделирования объектов с нестабильным функционированием и неопределенностью в данных. — Иркутск: Облинфорпечать, 1996. — 320 с.
10. **Носков С. И.** Построение эконометрических зависимостей с учетом критерия "согласованность поведения" // Кибернетика и системный анализ. — 1994. — № 1. — С. 177.

-
11. **Носков С. И.** Критерий "согласованность поведения" в регрессионном анализе // Современные технологии. Системный анализ. Моделирование. — 2013. — № 1 (37). — С. 107–110.
 12. **GRETl**. URL: <http://gretl.sourceforge.net/ru.html>.
 13. **Kreinovich V., Lakeyev A. V., Noskov S. I.** Approximate linear algebra is intractable // Linear Algebra and its Applications. — 1996. — Vol. 232, No. 1–3. — P. 45–54.
 14. **Лакеев А. В., Носков С. И.** Описание множества решений линейного уравнения с интервально заданными оператором и правой частью // Доклады РАН. — 1993. — Т. 330, № 3. — С. 430–433.
 15. **Golovchenko V. B., Noskov S. I.** Estimation of an econometric model using statistical data and expert information // Automation and Remote Control. — 1991. — № 4. — P. 123–132.
 16. **Базилевский М. П., Носков С. И.** Алгоритм построения линейно-мультиплексной регрессии // Современные технологии. Системный анализ. Моделирование. — 2011. — № 1 (29). — С. 88–92.
 17. **Базилевский М. П., Носков С. И.** Алгоритм формирования множества регрессионных моделей с помощью преобразования зависимой переменной // Международный журнал прикладных и фундаментальных исследований. — 2011. — № 3. — С. 159–160.
 18. **The Cyber Kill Chain**. URL: <https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html> (дата обращения 21.02.2020).
 19. **Новиков С. Н., Урманова Е. В.** Математическое моделирование нарушителя информационной безопасности // Современные проблемы телекоммуникаций. Материалы Российской научно-технической конференции. — 2019. — С. 94–97.
 20. **Егошин Н. С., Конев А. А., Шелупанов А. А.** Формирование модели нарушителя // Безопасность информационных технологий. — 2017. — Т. 24, № 4. — С. 19–26.
 21. **Kour R., Thaduri A., Karim R.** Railway Defender Kill Chain to Predict and Detect Cyber-Attacks // Journal of Cyber Security and Mobility. — 2019. — Vol. 9. — P. 47–90.
 22. **Hutchins E. M., Cloppert M. J., Amin R. M.** Intelligence-driven computer network defense informed by analysis of adversary campaign sand intrusion kill chains // Leading Issues in Information Warfare & Security Research. — 2011. — Vol. 1, No. 1. — P. 80.
 23. **Kim H., Kwon H., Kim K. K.** Modified cyber kill chain model for multimedia service environments // Multimedia Tools and Applications. — 2019. — Vol. 78, No. 3. — P. 3153–3170.
 24. **Ayrour Y., Raji A., Nassar M.** Modelling cyber-attacks: A survey study // Network Security. — 2018. — Vol. 2018, No. 3. — P. 13–19.

Regression Model of Structural Factors of Cyber Threats

S. I. Noskov, sergey.noskov.57@mail.ru, **A. S. Vergasov**, tluck@inbox.ru, Irkutsk State Transport University, Irkutsk, 664074, Russian Federation

Corresponding author:

Noskov Sergey I., Professor, Irkutsk State Transport University, 664074, Irkutsk, Russian Federation
E-mail: sergey.noskov.57@mail.ru

*Received on March 19, 2020
Accepted on June 08, 2020*

The work is based on a statistical analysis of data on cyberthreats classified by three groups of factors: motives of intruders, categories of victims, objects of attacks. This classification was adopted by Positive Technologies, which specializes in the development of information security systems and a comprehensive analysis of current trends in the implementation of cyber threats. As a result of processing the quarterly statistical information (from the 1st quarter of 2017 to 2nd quarter of 2019), an open regression recursive model of the dynamics of the nineteen structural factors of cyberthreats was constructed, which allows solving a wide range of problems of analysis and forecasting.

Keywords: regression model, cyberthreats, cyber kill chain, adequacy criteria, forecasting, recursion

For citation:

Noskov S. I., Vergasov A. S. Regression Model of Structural Factors of Cyber Threats, *Programmnaya Ingeneriya*, 2020, vol. 11, no. 4, pp. 251–256

DOI: 10.17587/prin.11.251-256

References

1. **High-profile** cases of cyber attacks and hacking in the world, available at: <https://ria.ru/20130809/955343983.html> (date of access 11/19/2019) (in Russian).
2. **Cyber** security loss, available at: http://www.tadviser.ru/index.php/Статья:Потери_организаций_от_киберпреступности (date of access 11/19/2019) (in Russian).
3. **Proshin A.** Hack in 60 seconds, *RBC*, available at: <https://rb.ru/opinion/trendy-kiberbezopasnost/> (date of access 11/19/2019) (in Russian).
4. **By the end** of 2016 cybercrime will cost \$650 billion to the global economy, available at: <https://www.croc.ru/news/detail/66901/> (date of access 11/19/2019) (in Russian).
5. **Loses** of companies from cyberattacks in the world in 2019 can reach \$2.5 trillion, available at: <https://www.kommersant.ru/doc/3957187/> (date of access 11/19/2019). (in Russian).

-
6. Draper N., Smith G. *Applied regression analysis. Multiple regression*. 3rd ed. Moscow, Dialectics, 2007, 912 p. (in Russian).
 7. Förster E., Rönz B. *Methods of correlation and regression analysis E*, Moscow, Finance and Statistics, 1981, 302 p. (in Russian).
 8. Radchenko S. G. *Regression analysis methodology*, Kiev, Kor niychuk, 2011, 376 p. (in Russian).
 9. Noskov S. I. *Technology for modeling objects with unstable operation and data uncertainty*, Irkutsk, Oblinforpechat', 1996, 320 p. (in Russian).
 10. Noskov S. I. Constructions of econometric relationships taking into account the criterion of "consistency of behavior", *Kibernetika i sistemnyi analiz*, 1994. no. 1, pp. 177 (in Russian).
 11. Noskov S. I. The criterion of "consistency of behavior" in the regression analysis, *Sovremennie tehnologii. Sistemnyi analiz. Modelirovanie*, 2013, no. 1 (37), pp. 107–110 (in Russian).
 12. GRETL, available at: <http://gretl.sourceforge.net/>
 13. Kreinovich V., Lakeyev A. V., Noskov S. I. Approximate linear algebra is intractable, *Linear Algebra and its Applications*, 1996, no. 1–3, pp. 45–54.
 14. Lakeyev A. V., Noskov N. I. A description of the set of solutions of a linear equation with interval defined operator and right-hand side, *Doklady Mathematics*, 1993, no. 3, pp. 518–523 (in Russian).
 15. Golovchenko V. B., Noskov S. I. Estimation of an econometric model using statistical data and expert information, *Automation and remote control*, 1991, no. 4, pp. 123–132 (in Russian).
 16. Bazilevsky M. P., Noskov S. I. Algorithm for constructing linear-multiplicative regression, *Sovremennie tehnologii*.
 17. Bazilevsky M. P., Noskov S. I. An algorithm for the formation of a multitude of regression models by transforming a dependent variable, *Mezdunarodniy zhurnal prikladnih i fundamental'nih issledovanij*, 2011, no. 3, pp. 159–160 (in Russian).
 18. The Cyber Kill Chain, available at: <https://www.lockheed-martin.com/en-us/capabilities/cyber/cyber-kill-chain.html> (date of access 11/19/2019).
 19. Novikov S. N., Urmanova E. V. Mathematical modeling of information security violator, *Sovremennie problem telekommunikaci. Materiali Rossiyskoy nauchno-tehnicheskoy konferencii*, 2019, pp. 94–97 (in Russian).
 20. Egoshin N. S., Konev A. A., Shelupanov A. A. The formation of the model of the intruder, *Bezopasnost' informacionih tehnologiy*, 2017, vol. 24, no. 4, pp. 19–26 (in Russian).
 21. Kour R., Thaduri A., Karim R. Railway Defender Kill Chain to Predict and Detect Cyber-Attacks, *Journal of Cyber Security and Mobility*, 2019, vol. 9, pp. 47–90.
 22. Hutchins E. M., Cloppert M. J., Amin R. M. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains, *Leading Issues in Information Warfare & Security Research*, 2011, vol. 1, no. 1, p. 80.
 23. Kim H., Kwon H., Kim K. K. Modified cyber kill chain model for multimedia service environments, *Multimedia Tools and Applications*, 2019, vol. 78, no. 3, pp. 3153–3170.
 24. Ayrour Y., Raji A., Nassar M. Modelling cyber-attacks: A survey study, *Network Security*, 2018, vol. 2018, no. 3, pp. 13–19.

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

Сдано в набор 10.06.2020 г. Подписано в печать 28.07.2020 г. Формат 60×88 1/8. Заказ Р1420
Цена свободная.

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

Рисунок к статье Е. В. Орловой

«МЕТОДЫ И МОДЕЛИ АНАЛИЗА ДАННЫХ И МАШИННОГО ОБУЧЕНИЯ В ЗАДАЧЕ УПРАВЛЕНИЯ ПРОИЗВОДИТЕЛЬСТВОМ ТРУДА»

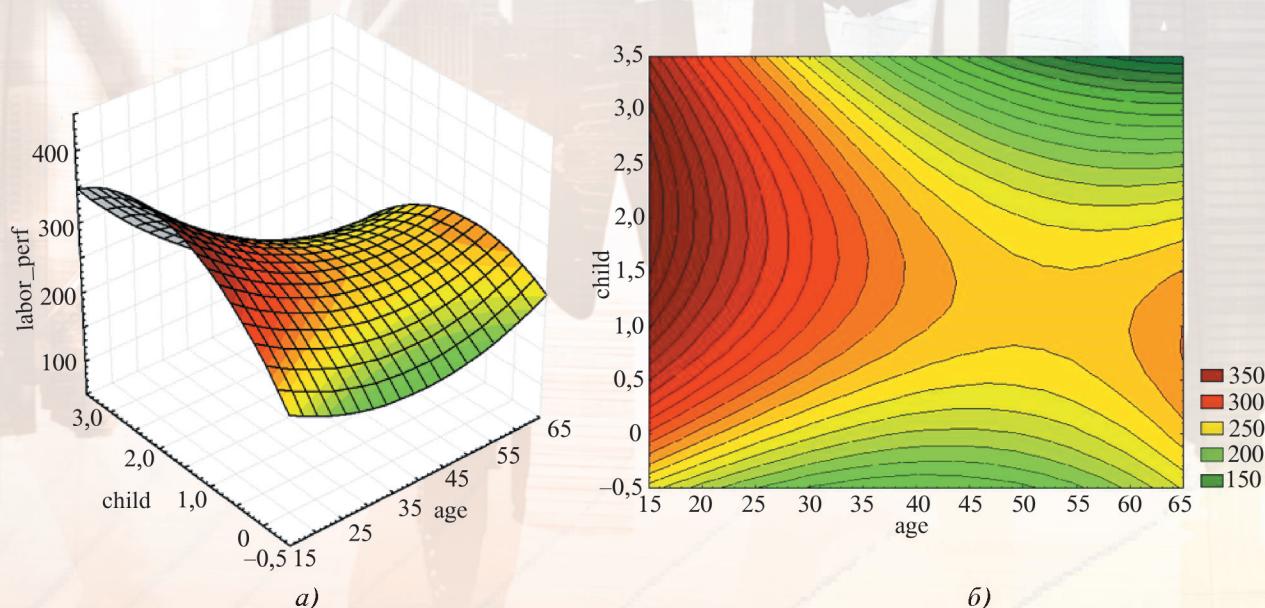


Рис. 3. Графики поверхности (а) и карты линий уровня (б)
переменной *labor_perf* в зависимости от факторов *age* и *child*

Рисунок к статье С. И. Носкова, А. С. Вергасова
«РЕГРЕССИОННАЯ МОДЕЛЬ СТРУКТУРНЫХ ФАКТОРОВ КИБЕРУГРОЗ»

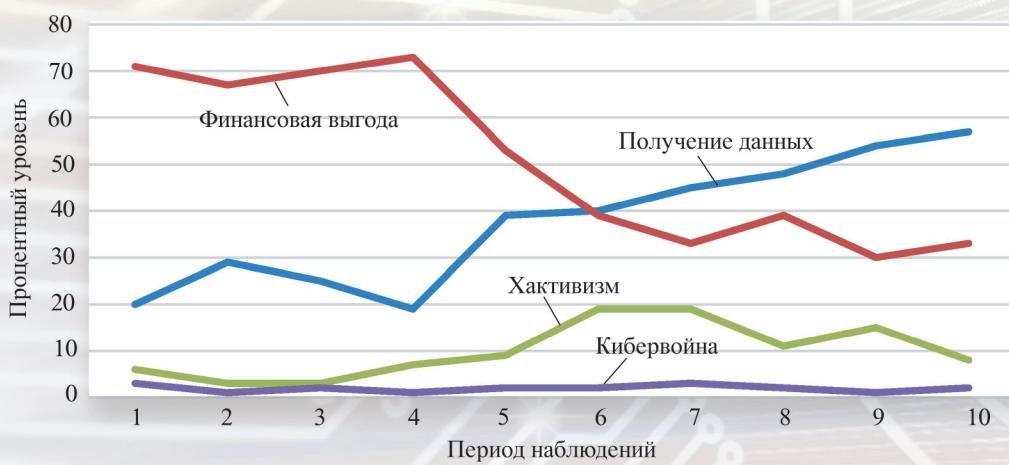
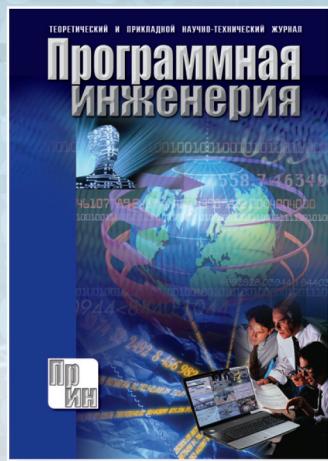


Рис. 1. Соотношение факторов в группе X_1

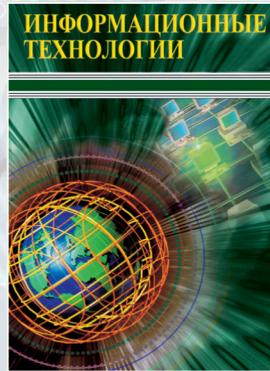
Издательство «НОВЫЕ ТЕХНОЛОГИИ» выпускает научно-технические журналы



Теоретический и прикладной научно-технический журнал **ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

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

Подписной индекс по Объединенному каталогу
«Пресса России» – 22765



Ежемесячный теоретический и прикладной научно- технический журнал **ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ**

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

Междисциплинарный теоретический и прикладной научно-технический журнал **НАНО- и МИКРОСИСТЕМНАЯ ТЕХНИКА**

В журнале освещаются современное состояние, тенденции и перспективы развитияnano- и микросистемной техники, рассматриваются вопросы разработки и внедрения nano- и микросистем в различные области науки, технологии и производства.



Подписной индекс по
Объединенному каталогу
«Пресса России» – 79493

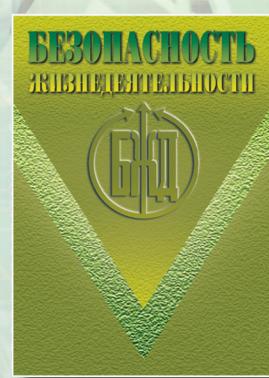


Ежемесячный теоретический и прикладной научно-технический журнал **МЕХАТРОНИКА, АВТОМАТИЗАЦИЯ, УПРАВЛЕНИЕ**

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

Научно-практический и учебно-методический журнал **БЕЗОПАСНОСТЬ ЖИЗНEDЕЯТЕЛЬНОСТИ**

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



Подписной индекс по
Объединенному каталогу
«Пресса России» – 79963

Подписной индекс по
Объединенному каталогу
«Пресса России» – 79492

Адрес редакции журналов для авторов и подписчиков:

107076, Москва, Строгинский пер., 4. Издательство "НОВЫЕ ТЕХНОЛОГИИ".
Тел.: (499) 269-55-10, 269-53-97. Факс: (499) 269-55-10. E-mail: antonov@novtex.ru