ISSN 2305-9052 (Print)
ISSN 2410-7034 (Online)

ВЕСТНИК ЮЖНО-УРАЛЬСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

BULLETIN OF THE SOUTH URAL STATE UNIVERSITY

СЕРИЯ

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА

2020, том 9, № 1

SERIES

COMPUTATIONAL MATHEMATICS AND SOFTWARE ENGINEERING

2020, volume 9, no. 1

1943

ВЕСТНИК

ЮЖНО-УРАЛЬСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2020
Т. 9, № 1

ISSN 2305-9052 (Print)
ISSN 2410-7034 (Online)

СЕРИЯ

«ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА»

Решением ВАК включен в Перечень научных изданий, в которых должны быть опубликованы результаты диссертаций на соискание ученых степеней кандидата и доктора наук

Учредитель — Федеральное государственное автономное образовательное учреждение высшего образования «Южно-Уральский государственный университет (национальный исследовательский университет)»

Тематика журнала:

- Вычислительная математика и численные методы
- Математическое программирование
- Распознавание образов
- Вычислительные методы линейной алгебры
- Решение обратных и некорректно поставленных задач
- Доказательные вычисления
- Численное решение дифференциальных и интегральных уравнений
- Исследование операций
- Теория игр
- Теория аппроксимации
- Информатика
- Искусственный интеллект и машинное обучение
- Системное программирование
- Перспективные многопроцессорные архитектуры
- Облачные вычисления
- Технология программирования
- Машинная графика
- Интернет-технологии
- Системы электронного обучения
- Технологии обработки баз данных и знаний
- Интеллектуальный анализ данных

BULLETIN

OF THE SOUTH URAL STATE UNIVERSITY

SERIES

"COMPUTATIONAL MATHEMATICS AND SOFTWARE ENGINEERING"

2020

Vol. 9, no. 1

ISSN 2305-9052 (Print)
ISSN 2410-7034 (Online)

Vestnik Yuzhno-Ural'skogo Gosudarstvennogo Universiteta.
Seriya "Vychislitel'naya Matematika i Informatika"

South Ural State University

The scope of the journal:

- Numerical analysis and methods
- Mathematical optimization
- Pattern recognition
- Numerical methods of linear algebra
- Reverse and ill-posed problems solution
- Computer-assisted proofs
- Numerical solutions of differential and integral equations
- Operations research
- Game theory
- Approximation theory
- Computer science
- Artificial intelligence and machine learning
- System software
- Advanced multiprocessor architectures
- Cloud computing
- Software engineering
- Computer graphics
- Internet technologies
- E-learning
- Database processing
- Data mining

Содержание

Contents

УДК 004.056.55, 512.54 **DOI: 10.14529/cmse200101**

РАЗРАБОТКА И РЕАЛИЗАЦИЯ ГРУППОВОГО ПРОТОКОЛА ГЕНЕРАЦИИ КЛЮЧА НА БАЗЕ IKE

© 2020 А.А. Волохов, Ю.В. Косолапов

Южный федеральный университет

(344066 Ростов-на-Дону, ул. им. Большая Садовая, д. 105/42)

E-mail: sashavolohov@yandex.ru, itaim@mail.ru

Поступила в редакцию: 16.07.2019

В качестве основы информационного взаимодействия участников в недоверенной среде часто выступает протокол выработки общего секретного ключа. С помощью такого ключа в дальнейшем может быть построен защищенный канал или защищенная сеть связи. В настоящее время актуальна задача разработки протоколов генерации общего ключа для группы участников. Одним из способов построения таких протоколов является обобщение протокола для двух участников на случай нескольких участников. В работе строится протокол генерации общего секретного ключа для группы участников (для конференции). В основе разработанного протокола лежит протокол IKE (Internet Key Exchange) из семейства протоколов IPSec для двух участников, обеспечивающий выполнение таких свойств безопасности, как аутентификация субъекта и сообщения, генерация новых ключей, защита от чтения назад, защита от повтора и ряда других. Стойкость разработанного протокола генерации ключа основана на сложности задачи дискретного логарифмирования в циклической группе. В работе исследуются свойства безопасности, обеспечиваемые построенным протоколом, в частности, исследуется стойкость к коалиционным атакам, актуальным для групповых протоколов. Также отмечаются некоторые особенности практического применения построенного протокола.

Ключевые слова: генерация секретного ключа, IKE, конференция.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Волохов А.А., Косолапов Ю.В. Разработка и реализация группового протокола генерации ключа на базе IKE // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 1. С. 5–19. DOI: 10.14529/cmse200101.

Введение

Развитие средств связи и совершенствование технологий коммуникации приводит к упрощению объединения различных субъектов связи в группы [1]. Например, средства видеонаблюдения, контроля доступа и охранной сигнализации объединяются в единую систему физической защиты помещений; устройства бытовой техники, объединенные с системой защиты помещений, формируют систему «умный дом»; видеокамеры и радары объединяются в систему «умный город»; ученые, эксперты, специалисты объединяются в группы для обсуждения интересующих вопросов; рядовые пользователи сети Интернет объединяются в группы для общения, игр и т.п. Часто каналы связи между участниками группы не защищены физически от пассивного или активного вмешательства, что создает угрозу конфиденциальности и целостности данным, передаваемым по каналам. Для нейтрализации этой угрозы могут применяться криптографические протоколы, базовым среди которых является протокол генерации общего для группы секретного ключа. Возможным способом построения протокола генерации ключа является обобщение использующихся на практике двухточечных протоколов (протоколов для двух участников). Такой подход обоснован, например, тем, что для использующихся на практике двухточечных протоколов известны обеспечиваемые этими протоколами свойства безопасности, сформулированные инженерным советом интернета IETF (Internet Engineering Task Force).

В настоящее время для защиты данных, передаваемых по сетевому протоколу IP (Internet Protocol), часто используется семейство двухточечных протоколов IPSec (IP security). В семейство протоколов IPSec кроме протоколов аутентификации сторон и шифрования передаваемых пакетов входит протокол генерации ключей IKE. В [2] отмечено, что протокол IKE из набора двадцати свойств безопасности, сформулированных организацией IETF, обеспечивает выполнение 10-ти свойств: G1-G3, G7, G9-G11, G13-G15.

В настоящей работе ставится задача построения и реализации группового протокола генерации ключей на основе протокола IKE. Кроме введения и заключения, работа содержит три раздела. Первый раздел посвящен обзору работ в области генерации групповых секретных ключей. Во втором разделе приводятся необходимые сведения о протоколе IKE и его стойкости. В третьем разделе на базе IKE строится групповой протокол генерации ключей, обосновывается его стойкость и рассматриваются свойства безопасности, обеспечиваемые этим протоколом в рамках модели угроз, предложенной в 1981 году Д. Долевым и А. Яо [3]. В конце второго раздела анализируется коммуникационная сложность протокола и рассматриваются особенности реализации разработанного протокола, в частности, проводится экспериментальная оценка времени генерации ключа конференцией в зависимости от числа участников.

1. Обзор работ

Способы генерации группового ключа можно разделить на сетевые протоколы и канальные протоколы. Канальные протоколы характеризуются тем, что для генерации ключа могут использоваться особенности среды передачи сигнала. В частности, в [4] и [5] предложен канальный протокол генерации общего секретного ключа для группы беспроводных устройств. Так как на канальном уровне для каждой пары устройств помеховая обстановка отличается от помеховой обстановки любой другой пары, то каждая такая пара на первом этапе может сгенерировать общий секретный ключ пары. Такие ключи далее используются на втором этапе для генерации общего ключа группы. Теоретические основы генерации в зашумленной среде общего ключа между парами участников заложены А. Вайнером в работе [6] в 1975 г.

В сетевых протоколах не доступна информация о помеховой обстановке, поэтому для генерации ключа используются иные криптографические примитивы. В работе [7] рассматривается случай, когда участники группы имеют общий пароль, и их задачей является генерация с помощью группового протокола Диффи—Хэллмана общего секретного ключа. Основным результатом [7] является доказательство стойкости группового протокола к атаке по словарю. В [8] групповой протокол Диффи—Хэллмана модифицируется с целью защиты от нарушителя, имеющего доступ к долговременным ключам. Для аутентификации участников группы в [9] разработан протокол генерации ключа, где групповой ключ широковещательно рассылается сервером, а для аутентификации участников группы используется схема разделения секрета Шамира.

Отметим, что в работах [7] и [8] участники протокола при генерации протокола взаимодействуют непосредственно друг с другом, а в [9] — через центральный сервер. Такие способы взаимодействия участников используются для относительно небольших групп. В случае большого числа участников возможно применение древовидной сети, где выделяются участники, ответственные за генерацию и передачу ключевого материала для других

субъектов. В [10] на основе протокола Диффи—Хэллмана строится протокол генерации ключа в децентрализованной древовидной сети; там же доказывается его стойкость.

В работе [11], с целью упрощения вычислений, групповой протокол генерации ключа строится на основе обобщенных полиномов Чебышева. При этом, как утверждается в [11] (без приведения доказательства), безопасность протокола не снижается по сравнению с протоколом на основе Диффи—Хэллмана. Отметим, что разработка доказуемо стойких криптографических протоколов, в том числе, протоколов генерации ключей, в основе стойкости которых не лежат задачи дискретного логарифмирования и факторизации, является актуальной в связи с развитием квантовых вычислений. К работам в этом направлении можно отнести работу [12], где строится схема распределения ключей на 3-дизайнах Адамара и доказывается стойкость разработанной схемы к атакам коалиций мощности не более двух.

Несмотря на актуальность разработки новых криптографических протоколов для групп участников (в частности, протоколов, стойких в постквантовую эпоху), в настоящее время, с целью упрощения перехода от двухточечных протоколов к групповым, актуальна задача доработки существующих двухточечных протоколов до групповых версий. В настоящей работе групповой протокол генерации ключа строится на основе применяемого на практике двухточечного протокола IKE.

2. Двухточечный протокол генерации ключа

2.1. Протокол Диффи—Хэллмана

Ядром протокола IKE является протокол Диффи—Хэллмана для выработки общего ключа в незащищенном от прослушивания канале связи [13]. В удобном виде приведем протокол Диффи—Хэллмана. Пусть $G = \langle g \rangle$ — группа порядка p, порожденная элементом g. При выполнении протокола обе взаимодействующие стороны, обозначаемые C_1 и C_2, выбирают большие случайные числа $i_1, i_2 (\in \mathbb{Z}_p)$ — секретные ключи Диффи—Хэллмана — и выполняют шаги, указанные ниже. Результатом выполнения этого протокола для двух

Протокол Диффи—Хеллмана

1: $C_1 \to C_2 : pkey_1 = g^{i_1}$
2: $C_2 \to C_1 : pkey_2 = g^{i_2}$
3: $C_1 : SK = pkey_{2,1} = (pkey_2)^{i_1}$
4: $C_2 : SK = pkey_{1,2} = (pkey_1)^{i_2}$

участников является общий ключ SK, который может быть, например, ключом для симметричного шифрования пересылаемых в дальнейшем сообщений. Числа $pkey_1$ и $pkey_2$ будем далее называть открытыми *полуключами* Диффи—Хэллмана.

Стойкость протокола Диффи—Хэллмана основана на том, что только по полуключам $pkey_1$ и $pkey_2$ вычислительно сложно найти ключ SK. Предположение о вычислительной сложности этой задачи также называется *предположением Диффи—Хэллмана* (Diffie-Hallman assumption, DH) или *предположением Диффи—Хэллмана о сложности вычисления* (Computational Diffie-Hallman assumption, CDH). Оценка сложности этой задачи тесно связана с оценкой вычислительной сложности задачи дискретного логарифмирования [14], когда по заданному $a \in G$ требуется найти целое неотрицательное решение x уравнения $g^x = a$ [13]. Предположение о сложности решения задачи дискретного логарифмирования называют *предположением о дискретном логарифме* (Discrete Logarithm assumption,

DL). Заметим, что если имеется эффективный алгоритм решения задачи дискретного логарифмирования, то имеется эффективный алгоритм нахождения ключа SK по полуключам $pkey_1$ и $pkey_2$. С другой стороны, если нет эффективного алгоритма для нахождения секретного ключа по открытым полуключам Диффи—Хэллмана, то нет эффективного алгоритма и для задачи вычисления дискретного логарифма. Иногда стойкость протокола Диффи—Хэллмана доказывается в рамках *предположения Диффи—Хэллмана о принятии решения* (Decisional Diffie-Hallman assumption, DDH) [15]. Согласно предположению DDH, случайные векторы $(g^{I_1}, g^{I_2}, g^{I_1 I_2})$ и $(g^{I_1}, g^{I_2}, g^{X})$ вычислительно *неразличимы* для достаточно большого числа p, где I_1, I_2, X — случайные величины, принимающие значения равновероятно из $\mathbb{Z}_p$.

Недостатком протокола Диффи—Хеллмана является отсутствие защиты от атаки «человек посередине», а также от засоряющей атаки, в ходе которой одна из сторон многократно отправляет свои полуключи. В частности, ни инициатор соединения, ни отвечающая сторона не могут достоверно определить, кем является их собеседник, что позволяет реализовывать атаку типа человек посередине. Для защиты от такой атаки может быть использована, например, взаимная аутентификация на основе асимметричных криптоалгоритмов (см. далее протокол IKE). При выполнении засоряющей атаки, например, стороной C_1, сторона C_2 должна многократно вычислять полуключи протокола, что может привести к отказу в обслуживании со стороны C_2. Чтобы предотвратить засоряющую атаку, к протоколу добавляются два раунда, в которых стороны обмениваются дополнительными наборами данных о клиентах, называемыми cookies. В общем случае, cookies представляют собой результат вычисления хеш-функции от уникальных идентификаторов, таких как IP-адрес, номер сетевого порта, номер протокола и т.п. Добавление cookies позволяет отвечающей стороне отсекать непрекращающиеся запросы от инициатора в начале протокола.

2.2. Протокол IKE

В упрощенном виде протокол IKE из семейства протоколов IPSec состоит из двух фаз, выполняемых друг за другом. Целью первой фазы является выработка для сеанса связи ключевого материала (см. (1)–(4)), на основе которого во второй фазе вырабатываются ключи шифрования сообщений, передаваемых в рамках сеанса (см. (9)). Обозначим $[\![m]\!]_{\mathcal{A}}$ шифрование сообщения m на открытом ключе $\mathcal{A}$ по алгоритму асимметричного шифрования, согласованному участниками протокола. Пусть $\mathcal{I}_j$ — открытый ключ участника C_j, $j = 1, 2$; $h : \mathcal{K} \times \{0,1\}^* \to \mathcal{H}_1$ — ключевая криптографическая хеш-функция с множеством ключей $\mathcal{K}$ и множеством значений $\mathcal{H}_1$, $\mathrm{hash} : \{0,1\}^* \to \mathcal{H}_2$ — бесключевая криптографическая хеш-функция с множеством значений $\mathcal{H}_2$. Фазы протокола IKE приведены в протоколах *IKE.Фаза 1* и *IKE.Фаза 2*.

На первом шаге (раунде) первой фазы инициатор C_1 отправляет заголовок HDR_1, содержащий, в том числе, *cookies* и идентификатор передаваемого сообщения Msg_{ID} (для второй фазы), а также предложенные параметры безопасности SA_{pr}, включающие алгоритмы и параметры симметричного и асимметричного шифрования, алгоритмы вычисления хеш-значений. Отвечающая сторона отправляет свой заголовок HDR_2 и выбранные параметры безопасности SA_s. Далее участники обмениваются случайными числами $N_j^{(1)}$, собственными идентификаторами ID_j и полуключами, шифруя передаваемые данные на открытых ключах получающей стороны с целью взаимной аутентификации (и защиты от атаки «человек посередине»). После четвертого раунда участники вычисляют ключевой

Протокол *IKE.Фаза 1*

$C_1 \to C_2 : HDR_1, SA_{pr}$
$C_2 \to C_1 : HDR_2, SA_s$
$C_1 \to C_2 : HDR_1, pkey_1, [\![N_1^{(1)}, ID_1]\!]_{\mathcal{I}_2}$
$C_2 \to C_1 : HDR_2, pkey_2, [\![N_2^{(1)}, ID_2]\!]_{\mathcal{I}_1}$
C_1, C_2 : вычисление S, S_d, S_a, S_e
$C_1 \to C_2 : HDR_1, [H_1]_{S_e}$

Протокол *IKE.Фаза 2*

1: $C_1 \to C_2 : HDR_1, [H_1^2, SA_s, N_1^{(2)}, pkey_1, ID_1, ID_2]_{S_e}$
2: $C_2 \to C_1 : HDR_2, [\tilde{H}_2^2, SA_s, N_2^{(2)}, pkey_2, ID_2, ID_1]_{S_e}$
3: $C_1 \to C_2 : HDR_1, [H^2]_{S_e}$

набор, состоящий из первоначального ключа S, ключа для создания других ключей S_d, ключа аутентификации S_a и ключа шифрования S_e:

$$S = h(\mathrm{hash}(\langle \textit{набор } N^{(1)} \rangle), \langle \textit{набор cookies} \rangle), \tag{1}$$

$$S_d = h(S, SK | \langle \textit{набор cookies} \rangle | 0), \tag{2}$$

$$S_a = h(S, S_d | SK | \langle \textit{набор cookies} \rangle | 1), \tag{3}$$

$$S_e = h(S, S_a | SK | \langle \textit{набор cookies} \rangle | 2). \tag{4}$$

Здесь $\langle$*набор* $N^{(1)}\rangle$ имеет вид $\langle N_1^{(1)} | N_2^{(1)} \rangle$, сгенерированный ключ SK равен $pkey_{1,2}$, а $\langle$*набор cookies*$\rangle$ равен $\langle cookies_1 | cookies_2 \rangle$. Здесь и далее $\mathbf{a}|\mathbf{b}$ обозначает конкатенацию векторов (строк) $\mathbf{a}$ и $\mathbf{b}$. Проверка правильности сгенерированного ключевого набора (аутентификация ключа) выполняется на шаге 6, где первым участником для $j = 1$ вычисляется значение

$$H_j = h(S, \langle \textit{набор полуключей, известных участнику } C_j \rangle | \langle \textit{набор cookies} \rangle | SA_s | ID_j), \tag{5}$$

которое шифруется с помощью сгенерированного ключа S_e (символом $[m]_a$ обозначается симметричное шифрование сообщения m на ключе a в рамках согласованной участниками симметричной криптосистемы) и отправляется второму участнику. Второй участник имеет необходимые данные для вычисления H_j (для $j = 1$) и проверки правильности общего ключа. Во второй фазе на первых двух шагах участники удостоверяются, что обе стороны используют один ключевой набор (1)–(4), сгенерированный на первой фазе, при этом:

$$H_j^2 = h(S_a, Msg_{ID} | SA_s | N_j^{(2)}), \tag{6}$$

$$\tilde{H}_j^2 = h(S_a, Msg_{ID} | N_j^{(1)} | SA_s | N_j^{(2)}), \tag{7}$$

$$H^2 = h(S_a, 0 | Msg_{ID} | \langle \textit{набор } N^{(1)} \rangle). \tag{8}$$

При этом для взаимной аутентификации участников используют случайные числа, которыми они обменялись в первой фазе. Взаимная аутентификация во второй фазе позволяет участникам убедиться, что собеседник не подменен. Третий шаг второй фазы используется для аутентификации ключевого материала, на основе которого генерируется общий ключ K шифрования сообщений в сеансе:

$$K = h(S_d, \langle \textit{набор } N^{(2)} \rangle). \tag{9}$$

3. Групповой протокол генерации ключа

Для построения группового протокола генерации ключа рассмотрим обобщение протокола Диффи—Хэллмана на случай $n(\in \mathbb{N})$ участников. Множество натуральных чисел от 1 до n обозначим $[n]$. В обобщенном протоколе генерируемый ключ SK имеет вид $g^{i_1,...,i_n}$, где i_j — секретный ключ Диффи—Хэллмана участника C_j, $j \in [n]$. Опишем процедуру генерации общего ключа. Пусть

$$pkey_{l_1,...,l_r} = g^{i_{l_1},...,i_{l_r}} \tag{10}$$

— полуключ порядка r, где числа $l_1, ..., l_r$ принадлежат множеству $[n]$ и попарно различны. Схема генерации общего ключа состоит в выполнении $n-1$ итераций: на r-ой итерации участники вычисляют и обмениваются полуключами r-ого порядка, $r \in [n-1]$ (для простоты полагается, что вместе с полуключом r-ого порядка передаются номера участников, на основе секретных ключей которых построен этот полуключ). При вычислении полуключей r-ого порядка используются полуключи $r-1$ порядка: участник C_j возводит в степень i_j все такие полученные на предыдущей итерации полуключи $pkey_{l_1,...,l_{r-1}}$ порядка $r-1$, для которых $j \notin \{l_1, ..., l_{r-1}\}$. Каждый полуключ $pkey_{l_1,...,l_r}$, вычисленный участником C_j на r-ой итерации, передается такому участнику C_k, для которого $k \notin \{l_1, ..., l_r\}$. Таким образом, на $(n-1)$-ой итерации каждый из участников отправит по одному полуключу порядка $n-1$, с помощью которого может быть найден общий ключ SK.

В рамках модели угроз Долева-Яо, пассивный наблюдатель перехватывает все полуключи всех порядков. В [16] доказано, что если протокол Диффи—Хэллмана для двух пользователей вычислительно стойкий в рамках предположения DDH, то случайные векторы

$$(g^{I_1}, ..., g^{I_n}, g^{I_1 I_2}, ..., g^{I_{n-1} I_n}, ..., g^{I_1 ... I_{n-1}}, ..., g^{I_2 ... I_n}, g^X), \tag{11}$$

$$(g^{I_1}, ..., g^{I_n}, g^{I_1 I_2}, ..., g^{I_{n-1} I_n}, ..., g^{I_1 ... I_{n-1}}, ..., g^{I_2 ... I_n}, g^{I_1 ... I_n}) \tag{12}$$

вычислительно неразличимы, когда случайные величины I_1,...,I_n, X принимают значения случайно и равновероятно из $\mathbb{Z}_p$. Это позволяет строить групповую версию протокола IKE.

3.1. Групповой протокол

В настоящей работе строится протокол с сервером, через который осуществляются пересылки сообщений, формируемых участниками в процессе генерации ключа. Ключом SK для многопользовательского протокола является значение $g^{i_1 ... i_n i_r}$, где g^{i_j} — полуключи участников для $j = 1, ..., n$, а g^{i_r} — полуключ сервера R. Значения $N^{(1)}_{sum} = \sum_{i=1}^{n} N^{(1)}_j$, $cookies_{hash} = \mathrm{hash}(cookies_1, ..., cookies_n)$ и $N^{(2)}_{sum} = \sum_{i=1}^{n} N^{(2)}_j$ используются при вычислении S, S_d, S_a, S_e, H_j, H^2 и K в формулах (1)–(5), (8) и (9) вместо соответственно $\langle$*набор* $N^{(1)}\rangle$, $\langle$*набор cookies*$\rangle$, $\langle$*набор* $N^{(2)}\rangle$. Пусть $N^{(i)}_{sum,j} = N^{(i)}_{sum} - N^{(i)}_j$ для $i \in [2]$, $\mathcal{R}$ — открытый ключ сервера R; для указания на то, что отправителем данных $\langle$данные$\rangle$ является сервер R, будем использовать обозначение $\langle$данные$\rangle_r$;

$$\hat{H}^{2,j}_r = h(S_a, Msg_{ID} | \tilde{N}^{(1)}_j | SA_s | N^{(2)}_{sum}). \tag{13}$$

В протоколе для $3 \le j \le n$ символом $pkey(C_{j-1})$ обозначен набор полуключей, полученных сервером от участника C_{j-1}, $pkey(C_1) = \{pkey_{1,r}, pkey_{n,r}\}$, а $pkey(C_{j-1}, j)$ — набор полуключей, получаемых путем возведения в степень i_j (секретный ключ Диффи—Хэллмана

участника C_j) полуключей из $pkey(C_{j-1})$: $pkey(C_{j-1}, j) = \{pkey^{i_j} : pkey \in pkey(C_{j-1})\}$. В частности, $pkey(C_j) = pkey(C_{j-1}, j)$ для $j = 2, ..., n$. Символом $\overline{pkey}(C_{n-1})$ обозначим следующий набор полуключей:

$$\overline{pkey}(C_{n-1}) = \{pkey_\tau : n \notin \tau, pkey_\tau \in \cup_{l=1}^{n-1} pkey(C_l)\},$$

а символом $\overline{pkey}(C_n)$ обозначим набор полуключей, отправляемых участником с номером n серверу (условие $pkey \neq pkey_{1,...,n-1,r}$ гарантирует, что участник C_n не отправит серверу общий ключ SK по открытому каналу):

$$\overline{pkey}(C_n) = \{(pkey)^{i_n} : pkey_\tau \in \overline{pkey}(C_{n-1}), pkey \neq pkey_{1,...,n-1,r}\}.$$

Протокол *Фаза 1*

1: $R \to C_j,\ j \in [n] : HDR_r, [\![SA_{pr}, N_j^{(1)}, ID_r]\!]_{\mathcal{I}_j}$
2: $C_j \to R,\ j \in [n] : HDR_j, h(N_j^{(1)}, ID_j), pkey_j, [\![SA_{s,j}, \tilde{N}_j^{(1)}, ID_j]\!]_{\mathcal{R}}$
3: $R \to C_2 : HDR_r, pkey_1, pkey(C_1), h(N_2^{(1)}, \tilde{N}_2^{(1)} | ID_2)$
4: $C_2 \to R : HDR_2, pkey_{1,2}, pkey(C_1, 2)$
5: ...
6: $R \to C_j : HDR_r, pkey_{1,...,j-1}, pkey(C_1), ..., pkey(C_{j-1}), h(N_j^{(1)}, \tilde{N}_j^{(1)} | ID_j)$
7: $C_j \to R : HDR_j, pkey_{1,...,j}, pkey(C_1, j), ..., pkey(C_{j-1}, j)$
8: ...
9: $R \to C_n : HDR_r, pkey_{1,...,n-1}, \overline{pkey}(C_{n-1}), h(N_n^{(1)}, \tilde{N}_n^{(1)} | ID_n)$;
10: $C_n \to R : HDR_n, pkey_{1,...,n}, \overline{pkey}(C_n)$;
11: $R \to C_j,\ j \in [n-1] : HDR_r, pkey_{1,2,...,j-1,j+1,...,n,r}, [\![N_{sum,j}^{(1)}, cookies_{hash}]\!]_{\mathcal{I}_j}$
12: $C_j, j \in [n]$: вычисление $S, S_d, S_a, S_e, h(H_j, \tilde{N}_j)$
13: $C_j \to R,\ j \in [n] : HDR_j, [h(H_j, \tilde{N}_j)]_{S_e}$

Протокол *Фаза 2*

1: $C_{init} \to R : [Msg_{ID}, ID_{init}, ID_r]_{S_e}$
2: $R \to C_j,\ j \in [n] : [Msg_{ID}, ID_r, ID_j]_{S_e}$
3: $C_j, j \in [n]$: вычисление H_j^2
4: $C_j \to R,\ j \in [n] : HDR_j, [\tilde{H}_j^2, SA_s, N_j^{(2)}, ID_j, ID_r]_{S_e}$
5: $R \to C_j,\ j \in [n] : HDR_r, [\hat{H}_r^{2,j}, SA_s, N_{sum,j}^{(2)}, ID_r, ID_j]_{S_e}$
6: $C_j, j \in [n]$: вычисление H^2, K
7: $C_j \to R,\ j \in [n] : HDR_j, [H^2, ID_j, ID_r]_{S_e}$

Сервер в первой фазе на шагах 1 и 11 выполняет по n пересылок, а на шагах 2–10 сервер участвует $n-1$ раз как инициатор пересылок. С другой стороны, каждый участник в первой фазе является инициатором пересылки три раза: шаги 2, 13 и один раз на одном из шагов с 3 по 12. Таким образом, суммарное число пересылок в первой фазе составляет порядка $6n$. На второй фазе сервер выполняет $2n$ пересылок (шаги 2 и 5), а участники совершают в совокупности $2n+1$ пересылок (шаги 1, 4 и 7). Общее количество пересылок составляет порядка $\mathcal{O}(8n)$. Заметим, что число пересылок и объем пересылаемых данных в разработанном протоколе существенно меньше, чем при широковещательной рассылке

участниками полуключей r-ого порядка, когда $r \in [n-1]$, так как в последнем случае требуется совершить порядка $\mathcal{O}(n^3)$ пересылок.

Для примера в табл. 1 приведены наборы пересылаемых полуключей при $n = 4$. В таблице выделены полуключи, на основе которых каждый участник C_j может вычислить общий секретный ключ SK, возведя полученные от сервера полуключи порядка n в степень, равную соответствующему секретному ключу i_j (так как в протоколе участвует сервер, то общий ключ будет порядка $n+1$).

Таблица 1

Наборы передаваемых полуключей для $n = 4$ на шагах 3–13 первой фазы протокола

Раунд протокола	Полуключи
$C_j \to R$, $j = 1, ..., n$	$pkey_j$
$R \to C_2$	$pkey_1$,$pkey_{1,r}$,$pkey_{4,r}$
$C_2 \to R$	$pkey_{1,2}$,$pkey_{1,2,r}$, $pkey_{2,4,r}$
$R \to C_3$	$pkey_{1,2}$,$pkey_{1,r}$,$pkey_{4,r}$, $pkey_{1,2,r}$, $pkey_{2,4,r}$
$C_3 \to R$	$pkey_{1,2,3}$, $pkey_{1,3,r}$,$pkey_{3,4,r}$, $pkey_{1,2,3,r}$, $pkey_{2,3,4,r}$
$R \to C_4$	$pkey_{1,2,3}$, $pkey_{1,r}$, $pkey_{1,2,r}$, $pkey_{1,3,r}$, $pkey_{1,2,3,r}$
$C_4 \to R$	$pkey_{1,2,3,4}$, $pkey_{1,4,r}$, $pkey_{1,2,4,r}$, $pkey_{1,3,4,r}$
$R \to C_1$	$pkey_{2,3,4,r}$
$R \to C_2$	$pkey_{1,3,4,r}$
$R \to C_3$	$pkey_{1,2,4,r}$

3.2. Анализ свойств безопасности группового протокола

В протоколе IKE предусмотрены опции, применение которых обеспечивает свойства безопасности G1-G3, G7, G9-G11, G13-G15. Ниже приводится анализ построенного протокола на предмет выполнения свойств безопасности. Напомним, что, согласно модели Долева-Яо, нарушитель может: 1) получить любое сообщение, которое передается по сети, 2) устанавливать соединение с любым другим пользователем от своего имени, 3) стать стороной, принимающей сообщения, 4) передавать сообщения от имени других пользователей. С другой стороны, нарушитель не может: 1) угадывать случайные числа из достаточно большого диапазона, 2) расшифровывать сообщения, не имея ключа, 3) найти секретный ключ по открытому ключу, 4) получить доступ к закрытым внутренним ресурсам средств связи, таким как оперативная память или жесткий диск других пользователей.

Утверждение 1. *Пусть $\mathcal{A}$ — множество возможных нарушителей, $R \notin \mathcal{A}$. Подмена сервера R до выполнения первой и/или до выполнения второй фазы может быть выявлена участниками протокола за полиномиальное время.*

Доказательство. Подмена сервера до выполнения первой фазы будет выявлена на третьем шаге первой фазы: для аутентификации серверу требуется расшифровать случайное число, отправленное каждым участником, и зашифрованное на открытом ключе сервера, чтобы отправить значение $h(N_j^{(1)}, \tilde{N}_j^{(1)} | ID_j)$. В рамках модели Долева-Яо, нарушитель не может по открытому ключу найти секретный ключ и угадывать случайные числа из достаточно большого диапазона, поэтому нарушитель не может подменить сервер до начала первой фазы. Во второй фазе для аутентификации сервера также используются случайные

числа $\tilde{N}_j^{(1)}$ (см. (13)), которые по каналам связи передаются только в зашифрованном на открытом ключе сервера виде. Так как вычисление значения хеш-функции и расшифрование выполняется за полиномиальное время, то и аутентификация сервера выполняется за полиномиальное время. □

Утверждение 2. *Пусть $\mathcal{A}$ — множество возможных нарушителей. Каждый нарушитель из $\mathcal{A}$ в рамках модели угроз Долева-Яо имеет доступ к полному набору полуключей, доступных легитимному участнику C_j для всех $j \in [n]$ после завершения первой фазы протокола.*

Доказательство. Доказательство следует из того, что все наборы полуключей в протоколе передаются в незашифрованном виде. □

Из утверждения 2 следует, что для аутентификации участников, то есть для доказательства субъектом соответствия своему идентификатору, набора полуключей для вычисления H_j не достаточно. Для аутентификации необходимо использовать информацию, доступную только самим участникам. Такой информацией может быть, например, случайное число $\tilde{N}_j^{(1)}$, сгенерированное участником на шаге 2 первой фазы для дальнейшей аутентификации сервера (в рамках модели Долева-Яо, нарушитель не может угадывать случайные числа из достаточно большого диапазона). Поэтому в протоколе на шаге 12 вычисляется хеш-значение $h(H_j, \tilde{N}_j^{(1)})$ (так как число $\tilde{N}_j^{(1)}$ известно только участнику C_j и серверу R).

Утверждение 3. *Пусть $\mathcal{P} = \{C_1, ..., C_n, R\}$ — множество легитимных участников протокола генерации группового ключа, $\mathcal{A}$ — множество нарушителей, $R \notin \mathcal{A}$. Любой нарушитель из множества $\mathcal{A}$, выдающий себя в первой фазе протокола за легитимного участника из множества $\mathcal{P}$, будет выявлен на одном из шагов первой фазы.*

Доказательство. Пусть $A(\in \mathcal{A})$ — нарушитель. Рассмотрим два случая: $A \in \mathcal{A} \backslash \mathcal{P}$ и $A \in \mathcal{P}$. В первом случае нарушитель не является легитимным участником, однако в рамках модели Долева-Яо, ему неизвестны секретные ключи участников. Тогда нарушитель, выдающий себя за участника C_j в начале первой фазы, будет выявлен на шаге 2, так как не сможет расшифровать случайное число $N_j^{(1)}$, отправленное сервером. Заметим, что нарушитель может попытаться выдать себя за участника C_j на одном из шагов 2–10. На этих шагах сервером не проверяется источник сообщений, поэтому нарушитель не будет выявлен. И в этом случае ключ Диффи—Хэллмана SK' будет известен нарушителю, а, следовательно, могут быть вычислены ключи S', S'_d, S'_a и S'_e (здесь штрихом обозначены ключи, которые получены с помощью секретного ключа Диффи—Хэллмана i'_j, выбранного нарушителем A). Однако на шаге 13 сервер проверяет правильность ключа, используя значение $h(H_j, \tilde{N}_j^{(1)})$. Так как случайное число $\tilde{N}_j^{(1)}$ известно только легитимному участнику (и не может быть угадано нарушителем), аутентифицированному к этому моменту сервером на шагах 1 и 2, то проверку на шаге 13 может пройти только легитимный участник. Аналогично показывается, что нарушитель из $\mathcal{P}$ также будет выявлен на шаге 2 или 13. □

Утверждение 4. *Пусть $\mathcal{P} = \{C_1, ..., C_n, R\}$ — множество легитимных участников протокола генерации группового ключа, $\mathcal{A}$ — множество нарушителей, $R \notin \mathcal{A}$. Любой нарушитель из множества $\mathcal{A}$, выдающий себя во второй фазе протокола за легитимного участника из множества $\mathcal{P}$, будет выявлен на одном из шагов второй фазы.*

Доказательство. Предполагается, что к началу второй фазы участники аутентифицированы, следовательно серверу известны случайные числа $\tilde{N}_1^{(1)},...,\tilde{N}_n^{(1)}$, сгенерированные в первой фазе участниками, а участникам — случайные числа $N_1^{(1)},...,N_n^{(1)}$, сгенерированные в первой фазе сервером. В рамках модели угроз Долева-Яо нарушитель не имеет доступа к этим случайным числам, поэтому использование во второй фазе случайных чисел с первой фазы обеспечивает взаимную аутентификацию сторон. Взаимная аутентификация во второй фазе выполняется с помощью хеш-значений $\tilde{H}_j^2$ и $\hat{H}_r^{2,j}$. □

Из утверждений 1, 3 и 4 следует, что для первой и второй фазы группового протокола выполняется свойство G1 — аутентификация субъекта.

Для аутентификации передаваемых в ходе протокола сообщений используются хеш-значения H_j (шаг 13 в первой фазе) и H^2 (шаг 7 во второй фазе), которые вычисляются с помощью криптографической хеш-функции с использованием случайных чисел, известных только серверу и легитимным участникам. Таким образом, выполняется свойство G2 — аутентификация сообщений. Использование случайных чисел, как в первой фазе, так и во второй фазе обеспечивает также защиту от повтора (свойство G3), защиту от чтения назад (свойство G9). Защита от чтения назад обеспечивается за счет того, что компрометация ключа SK не позволяет в случае использования криптографической хеш-функции h найти ключевой материал (1)–(4), зависящий не только от SK, но и от случайных чисел, передаваемых между клиентами и сервером в зашифрованном виде. Применение случайных чисел в каждой фазе обеспечивает формирование новых ключей (свойство G10). Однако во второй фазе не выполняется свойство совершенной прямой секретности (PFS, Perfect Forward Security), так как во второй фазе ключи генерируются на основе ключа Диффи—Хэллмана, полученного в первой фазе. Генерация во второй фазе ключей Диффи—Хэллмана, как показали эксперименты (см. раздел 3.3), существенно замедляет скорость обмена сообщениями: с ростом числа участников время генерации общего ключа растет нелинейно. Отметим, что в двухточечном протоколе IKE предусмотрен режим, обеспечивающий PFS.

Утверждение 5. *Пусть $\mathcal{P} = \{C_1, ..., C_n, R\}$ — множество легитимных участников протокола генерации группового ключа, $\mathcal{A}$ — множество нарушителей, $R \notin \mathcal{A}$. В завершении первой и/или второй фазы сервер либо обнаруживает вмешательство нарушителя, либо фиксирует, что только участникам из $\mathcal{P}$ известен секретный ключ.*

Доказательство. Из утверждений 1, 3 и 4 вытекает, что нарушитель может быть выявлен в течении первой или второй фазы. Неизвестность секретного ключа вытекает из стойкости группового протокола Диффи—Хэллмана в рамках предположения DDH (наборы (11) и (12) неотличимы за полиномиальное время). □

Из утверждения 5 вытекает свойство G7: сервер получает подтверждение того, что никакой другой участник, кроме заранее определенных участников конференции, не может получить доступа ни к одному секретному ключу.

Передача ассоциации безопасности SA_{pr} и SA_s в шифрованном виде (см. шаги 1 и 2 первой фазы) обеспечивает выполнение свойства G11 — стороны могут безопасно договориться о параметрах защищенной связи. Шифрование идентификаторов (ID) позволяет обеспечить свойство G13 — анонимность при прослушивании, а так как участники не подписывают сообщения электронной цифровой подписью, и идентификаторы, в общем случае, могут быть псевдонимами, меняющимися от сеанса к сеансу, то обеспечивается частичная

анонимность при работе с другими участниками (свойство G14). Частичная анонимность обеспечивается в том смысле, что серверу известны публичные ключи участников; с другой стороны, всем участникам известен идентификатор сервера, так как на втором шаге первой фазы участниками используется публичный ключ сервера. В протоколе предусмотрена ограниченная защищенность от атак типа отказ в обслуживании (свойство G15): вычисления общего секретного ключа по групповому протоколу Диффи—Хэллмана происходят после аутентификации клиентов (шаги 1 и 2), а также за счет *cookies*, передаваемых в заголовках сообщений. Частичная защита связана с тем, что ряд атак может проводиться после этапа аутентификации субъектов, например, подмена легитимного участника нарушителем.

Заметим, что из утверждений 1, 3, 4 и 5 вытекает необходимость в доверии серверу R, так как выполнение свойств G1 и G7 доказано при допущении, что сервер не входит в число нарушителей. Недоверенный сервер может выдать себя за любого участника конференции и посылать сообщения от его имени. В этом случае свойства G1 и G7 не выполняются: подмена сервером одного участника (или нескольких) останется незамеченной другими участниками конференции.

3.3. Особенности реализации

В работе для реализации пользовательской части использовался фреймворк Vue, который позволяет применять реактивные данные на языке JavaScript для удобного взаимодействия с пользователем. Серверная часть реализована с помощью сервера Node.js, фреймворка Express и пакета Cors, который позволяет абстрагироваться от низкоуровневой архитектуры программы и описывать непосредственно логику хранения и пересылок данных. Вычисления проводились на MacBook Pro (13-inch, 2018, Four Thunderbolt 3 Ports), процессор 2,3 GHz Intel Core i5, 8 ГБ 2133 MHz LPDDR3, видеокарта Intel Iris Plus Graphics 655 1536 МБ. После выполнения протокола у каждого из клиентов на странице браузера отображается итоговый ключ K, вычисляемый в соответствии с (9) и используемый для дальнейшего шифрования сообщений. При реализации были проведены замеры времени между первым отправляемым сообщением и выработкой ключа K на стороне участника с номером n, а также на стороне сервера между первым сообщением получаемым от участника с номером 1 и последним сообщением, отправляемым участнику с номером n. Для реализации группового протокола Диффи—Хэллмана выбрана мультипликативная группа конечного поля $\mathbb{F}_{2^{256}}$, а для работы с большими числами использована библиотека BigInteger.

Несмотря на то, что разработанный протокол с сервером ($\mathcal{O}(8n)$ пересылок) обладает меньшей коммуникативной сложностью, чем протокол без сервера ($\mathcal{O}(n^3)$ пересылок), использование тяжелой арифметики больших чисел в протоколе Диффи—Хэллмана приводит к тому, что с ростом числа участников конференции время генерации ключа растет нелинейно. Как видно из табл. 2, даже при небольшой длине ключа Диффи—Хэллмана для 14 клиентов ключ генерируется, без учета времени на пересылку, порядка 10 с.

Заключение

В работе на основе двухточечного протокола IKE построен протокол генерации ключа для группы участников. Представляется, что в разработанном протоколе может быть использован не только протокол Диффи—Хэллмана, но и другие криптографические примитивы, такие как полиномы Чебышева, а также примитивы на основе помехоустойчивых

Таблица 2

Время выполнения от количества клиентов n, мс. Длина ключ Диффи—Хэллмана — 256 бит

n	**На сервере**	**На клиенте**	n	**На сервере**	**На клиенте**
2	888	915	9	724	2189
3	701	2023	10	1947	1982
4	333	1557	11	2260	2299
5	938	2228	12	2528	3830
6	881	2340	13	4977	5021
7	644	1754	14	9130	9180
8	886	1980			

кодов. Отметим, что использование арифметики больших чисел приводит к низкой скорости генерации ключа даже для относительно небольших групп (см. табл. 2). Кроме того, согласно [17], для задачи дискретного логарифмирования имеется эффективный алгоритм для квантовых компьютеров. Поэтому криптографические алгоритмы, основанные, в частности, на предположениях DL, DH или DDH, в постквантовую эпоху не будут обеспечивать высокой стойкости. В связи с этим особенно актуальна адаптация разработанного протокола для применения в нем криптографических примитивов на основе некоторых кодовых криптосистем типа Мак-Элиса, обзор которых приведен, например, в [18].

Для разработанного протокола показано, что выполнение ряда свойств безопасности, которые обеспечивает протокол IKE, требует наличие доверия к серверу, через который происходит общение участников при генерации ключа. В случае использования недоверенного сервера выполнение, как минимум, свойств G1 (аутентификация субъекта) и G7 (гарантия неизвестности ключа нелегитимным участникам) не гарантируется. В связи с этим еще одной актуальной задачей для дальнейшего исследования является доработка протокола для использования недоверенного сервера.

Литература

1. Bilal M., Kang S.-G. A Secure Key Agreement Protocol for Dynamic Group // Journal Cluster Computing. 2017. Vol. 20, no. 3. P. 2779–2792. DOI: 10.1007/s10586-017-0853-0.

2. Черемушкин А.В. Криптографические протоколы: основные свойства и уязвимости // Прикладная дискретная математика. 2009. Приложение 2. С. 115–150.

3. Dolev D., Yao A.C. On the Security of Public Key Protocol // IEEE Transactions on Information Theory. 1983. Vol. 29, no. 2. P. 198–208. DOI: 10.1109/tit.1983.1056650.

4. Liu H., Yang J., Wang Y., Chen Y.J., Koksal C.E. Group Secret Key Generation via Received Signal Strength: Protocols, Achievable Rates, and Implementation // IEEE Transactions on Mobile Computing. 2014. Vol. 13, no. 12. P. 2820–2835. DOI: 10.1109/TMC.2014.2310747.

5. Xu P., Cumanan K., Ding Z., Dai X., Leung K.K. Group Secret Key Generation in Wireless Networks: Algorithms and Rate Optimization // IEEE Transactions on Information Forensics and Security. 2016. Vol. 11, no. 8. P. 1831–1846. DOI: 10.1109/TIFS.2016.2553643.

6. Wyner A.D. The wire-tap channel // The Bell System Technical Journal. 1975. Vol. 54, no. 8. P. 1355–1387. DOI: 10.1002/j.1538-7305.1975.tb02040.x.

7. Bresson E., Chevassut O., Pointcheval D. Group Diffie-Hellman Key Exchange Secure against Dictionary Attacks // 8th International Conference on the Theory and Application of Cryptology and Information Security (Queenstown, New Zealand, December, 1–5, 2002). Lecture Notes in Computer Science. 2002. P. 497–514. DOI: 10.1007/3-540-36178-2_31.

8. Bresson E., Manulis M. Securing Group Key Exchange against Strong Corruptions and Key Registration Attacks // International Journal of Applied Cryptography. 2008. Vol. 1, no. 2. P. 91–107. DOI: 10.1504/IJACT.2008.021083.

9. Baiju B.V. Secret Key Sharing Scheme Based On Key Generation Centre For Authenticated Exchange Of Messages // International Journal of Engineering Science Invention. 2013. Vol. 2, no. 11. P. 15–21.

10. Kim Y., Perrig A., Tsudik G. Tree-based Group Key Agreement // ACM Transactions on Information and System Security. 2004. Vol. 7, no. 1. P. 60–96. DOI: 10.1145/984334.984337.

11. Lin T.-H., Tsung C.-K., Lee T.-F., Wang Z.-B. A Round-Efficient Authenticated Key Agreement Scheme Based on Extended Chaotic Maps for Group Cloud Meeting // Sensors. 2017. Vol. 17, no. 12. P. 1–14. DOI: 10.3390/s17122793.

12. Деундяк В.М., Таран А.А. Система распределения ключей на дизайнах. // Моделирование и анализ информационных систем. 2019. Т. 26, № 2. С. 229–243. DOI: 10.18255/1818-1015-2019-2-229-243.

13. Diffie W., Hellman M.E. New Directions in Cryptography // IEEE Transactions on Information Theory. 1976. Vol. 22, no. 6. P. 644–654. DOI: 10.1109/TIT.1976.1055638.

14. ElGamal T. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms // IEEE Transactions on Information Theory. 1985. Vol. 31, no. 4. P. 469–472. DOI: 10.1109/TIT.1985.1057074.

15. Boneh D. The Decision Diffie–Hellman Problem // Third International Symposiun, ANTS-III (Portland, Oregon, USA, June, 21–25, 1998). Lecture Notes in Computer Science. 1998. Vol. 1423. P. 48–63. DOI: 10.1007/BFb0054851.

16. Steiner M., Tsudik G., Waidner M. Diffie-Hellman Key Distribution Extended to Group Communication // 3rd ACM conference on Computer and communications security (New Delhi, India, March, 14–15, 1996). New York, ACM. 1996. P. 31–37. DOI: 10.1145/238168.238182.

17. Sendrier N. Code-Based Cryptography: State of the Art and Perspectives // IEEE Security & Privacy. 2017. Vol. 15, no. 4. P. 44–50. DOI: 10.1109/MSP.2017.3151345.

18. Deundyak V.M., Kosolapov Yu.V. On the Berger–Loidreau Cryptosystem on the Tensor Product of Codes // Journal of Computational and Engineering Mathematics. 2018. Vol. 5, no. 2. P. 16–33. DOI: 10.14529/jcem180202.

Волохов Александр Александрович, магистрант, Южный федеральный университет, Институт математики, механики и компьютерных наук им. И.И. Воровича (Ростов-на-Дону, Российская Федерация)

Косолапов Юрий Владимирович, к.т.н., кафедра алгебры и дискретной математики, Южный федеральный университет, Институт математики, механики и компьютерных наук им. И.И. Воровича (Ростов-на-Дону, Российская Федерация)

DOI: 10.14529/cmse200101

DEVELOPMENT AND IMPLEMENTATION OF THE CONFERENCE SECRET KEY GENERATION PROTOCOL BASED ON IKE

© 2020 A.A. Volokhov, Y.V. Kosolapov

Southern Federal University

(Bolshaya Sadovaya 105/42, Rostov-on-Don, 344006 Russia)

E-mail: sashavolohov@yandex.ru, itaim@mail.ru

Received: 16.07.2019

The protocol for generating a shared secret key often acts as the basis for informational interaction of participants in an untrusted environment. With the help of such a key, a secure channel or a secure communication network can be built in further interactions. Currently, the task of developing protocols for generating a shared key for a group of participants is relevant. One way to build such protocols is to generalize the protocol for two participants to the case of several participants. In the paper a protocol for generating a shared secret key for a group of participants (for a conference) is developed. The developed protocol is based on the Internet Key Exchange (IKE) protocol from the IPSec family of protocols for two participants, which ensures the implementation of security properties, such as authentication of the subject and message, generation of new keys, protection against reading back, protection against repetition, and a number of others. The strength of the developed key generation protocol is based on the complexity of the discrete logarithm problem in a cyclic group. The work studies the security properties provided by the constructed protocol, in particular, it studies the resistance to coalition attacks that are relevant for group protocols. Some features of the practical application of the constructed protocol are also noted.

Keywords: private key generation, IKE, conference.

FOR CITATION

Volokhov A.A., Kosolapov Y.V. Development and Implementation of the Conference Secret Key Generation Protocol Based on IKE. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2020. Vol. 9, no. 1. P. 5–19. (in Russian) DOI: 10.14529/cmse200101.

References

1. Bilal M., Kang S.-G. A Secure Key Agreement Protocol for Dynamic Group. Journal Cluster Computing. 2017. Vol. 20, no. 3. P. 2779–2792. DOI: 10.1007/s10586-017-0853-0.
2. Cheremushkin A.V. Cryptographic Protocols: Basic Properties and Vulnerabilities. Applied discrete mathematics. Appendix. 2009. no. 2. P. 115–150. (in Russian)
3. Dolev D., Yao A.C. On the security of public key protocol. IEEE Transactions on Information Theory. 1983. Vol. 29, no. 2. P. 198–208. DOI: 10.1109/tit.1983.1056650.
4. Liu H., Yang J., Wang Y., Chen Y. J., Koksal C.E. Group Secret Key Generation via Received Signal Strength: Protocols, Achievable Rates, and Implementation. IEEE Transactions on Mobile Computing. 2014. Vol. 13, no. 12. P. 2820–2835. DOI: 10.1109/TMC.2014.2310747.
5. Xu P., Cumanan K., Ding Z., Dai X., Leung K.K. Group Secret Key Generation in Wireless

Networks: Algorithms and Rate Optimization. IEEE Transactions on Information Forensics and Security. 2016. Vol. 11, no. 8. P. 1831–1846. DOI: 10.1109/TIFS.2016.2553643.

6. Wyner A.D. The Wire-tap Channel. The Bell System Technical Journal. 1975. Vol. 54, no. 8. P. 1355–1387. DOI: 10.1002/j.1538-7305.1975.tb02040.x.
7. Bresson E., Chevassut O., Pointcheval D. Group Diffie-Hellman Key Exchange Secure against Dictionary Attacks. 8th International Conference on the Theory and Application of Cryptology and Information Security (Queenstown, New Zealand, December, 1–5, 2002). Lecture Notes in Computer Science. 2002. P. 497–514. DOI: 10.1007/3-540-36178-2_31.
8. Bresson E., Manulis M. Securing Group Key Exchange against Strong Corruptions and Key Registration Attacks. International Journal of Applied Cryptography. 2008. Vol. 1, no. 2. P. 91–107. DOI: 10.1504/IJACT.2008.021083.
9. Baiju B.V. Secret Key Sharing Scheme Based On Key Generation Centre For Authenticated Exchange Of Messages. International Journal of Engineering Science Invention. 2013. Vol. 2, no. 11. P. 15–21.
10. Kim Y., Perrig A., Tsudik G. Tree-based Group Key Agreement. ACM Transactions on Information and System Security. 2004. Vol. 7, no. 1. P. 60–96. DOI: 10.1145/984334.984337.
11. Lin T.-H. , Tsung C.-K., Lee T.-F., Wang Z.-B. A Round-Efficient Authenticated Key Agreement Scheme Based on Extended Chaotic Maps for Group Cloud Meeting. Sensors. 2017. Vol. 17, no. 12. P. 1–14. DOI: 10.3390/s17122793.
12. Deundyak V.M., Taran A.A. Key Distribution System Based on Hadamard Designs. Modeling and Analysis of Information Systems. 2019. Vol. 26, no. 2. P. 229–243. (in Russian) DOI: 10.18255/1818-1015-2019-2-229-243.
13. Diffie W., Hellman M.E. New Directions in Cryptography. IEEE Transactions on Information Theory. 1976. Vol. 22, no. 6. P. 644–654. DOI: 10.1109/TIT.1976.1055638.
14. ElGamal T. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory. 1985. Vol. 31, no. 4. P. 469–472. DOI: 10.1109/TIT.1985.1057074.
15. Boneh D. The Decision Diffie–Hellman Problem. Third International Symposiun, ANTS-III (Portland, Oregon, USA, June, 21–25, 1998). Lecture Notes in Computer Science. 1998. Vol. 1423. P. 48–63. DOI: 10.1007/BFb0054851.
16. Steiner M., Tsudik G., Waidner M. Diffie-Hellman key distribution extended to group communication. 3rd ACM conference on Computer and communications security (New Delhi, India, March, 14–15, 1996). New York, ACM. 1996. P. 31–37. DOI: 10.1145/238168.238182.
17. Sendrier N. Code-Based Cryptography: State of the Art and Perspectives. IEEE Security & Privacy. 2017. Vol. 15, no. 4. P. 44–50. DOI: 10.1109/MSP.2017.3151345.
18. Deundyak V.M., Kosolapov Yu.V. On the Berger–Loidreau Cryptosystem on the Tensor Product of Codes. Journal of Computational and Engineering Mathematics. 2018. Vol. 5, no. 2. P. 16–33. DOI: 10.14529/jcem180202.

УДК 004.72, 004.75, 004.382.2 DOI: 10.14529/cmse200102

ОБ ИСПОЛЬЗОВАНИИ ФЕДЕРАЛЬНОЙ НАУЧНОЙ ТЕЛЕКОММУНИКАЦИОННОЙ ИНФРАСТРУКТУРЫ ДЛЯ СУПЕРКОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЙ*

© 2020 Г.И. Савин, Б.М. Шабанов, А.В. Баранов, А.П. Овсянников, А.А. Гончар

Межведомственный суперкомпьютерный центр РАН — филиал ФГУ ФНЦ НИИСИ РАН (119334 Москва, Ленинский пр., д. 32а)

E-mail: savin@jscc.ru, shabanov@jscc.ru, antbar@mail.ru, ovsiannikov@jscc.ru, andrey.gonchar@jscc.ru

Поступила в редакцию: 09.01.2020

Статья посвящена перспективам развития научной телекоммуникационной инфраструктуры на базе национальной исследовательской компьютерной сети нового поколения (НИКС), образованной путем интеграции ведомственных научно-образовательных сетей RUNNet и RASNet. Показаны возможности новой сети для объединения и организации взаимодействия суперкомпьютерных ресурсов и обеспечения безбарьерного доступа к ним. На основе обобщенного мирового опыта показано, что суперкомпьютерные инфраструктуры предъявляют специальные требования к телекоммуникационной сети по передаче данных и наличию ряда дополнительных сервисов. Эти требования выходят далеко за рамки услуг коммерческих операторов связи и, как правило, могут быть удовлетворены только объединенными усилиями национальных научно-образовательных сетей.

Рассмотрены ключевые элементы федеральной телекоммуникационной инфраструктуры, необходимые для объединения высокопроизводительных вычислительных ресурсов: высокопроизводительные каналы связи с заданным качеством обслуживания, их автоматическое выделение по требованию и по расписанию, доверенная сетевая среда, федеративная аутентификация и авторизация, обеспечение надежности и безопасности, сквозной мониторинг пути передачи данных между конечными пользователями. На основе анализа жизненного цикла суперкомпьютерного задания, мигрирующего в сети суперкомпьютерных центров коллективного пользования (СКЦ), сформулированы требования к телекоммуникационной инфраструктуре НИКС и сервисам на ее основе со стороны распределенной сети СКЦ.

Ключевые слова: национальная сеть науки и образования, суперкомпьютерный центр, центр коллективного пользования, распределенные вычисления, телекоммуникационная инфраструктура.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Савин Г.И., Шабанов Б.М., Баранов А.В., Овсянников А.П., Гончар А.А. Об использовании федеральной научной телекоммуникационной инфраструктуры для суперкомпьютерных вычислений // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 1. С. 20–35. DOI: 10.14529/cmse200102.

Введение

Наличие собственной надежной и высокопроизводительной телекоммуникационной инфраструктуры для науки и образования, обеспечивающей доступ к высокопроизводительным вычислительным ресурсам, суперкомпьютерным центрам, их сетевое взаимодействие, является сегодня необходимым условием для выхода на мировой уровень достижений в самых различных областях науки и технологий.

*Статья рекомендована к публикации программным комитетом международной конференции «Суперкомпьютерные дни в России — 2019».

Как отмечалось в [1], инфраструктура суперкомпьютерных центров включает не только вычислительную технику различной архитектуры, но и средства визуализации, информационные ресурсы и систему телекоммуникаций, причем приоритетное значение придается развитию систем обеспечения удаленного доступа со сбалансированной иерархией сетей различной пропускной способности.

Удаленный доступ к ресурсам суперкомпьютерного центра коллективного пользования (СКЦ) осуществляется как по выделенным сетям с высокой пропускной способностью (десятки гигабит в секунду), так и через публичные сети. Первые позволяют передавать значительные объемы данных между системами хранения данных (СХД) организаций-пользователей и СХД СКЦ, вторые — запускать задания, используя данные, заранее размещенные в СХД СКЦ, и там же сохранять результаты вычислений.

Мировой опыт в области создания и развития научно-образовательных телекоммуникационных сетей свидетельствует об особой роли в организации информационного обмена СКЦ национальных научно-образовательных сетей (или национальных сетей науки и образования — National Research and Education Network, NREN).

Национальные научно-образовательные сети эволюционируют, откликаясь на возрастающие потребности науки и образования, оперативно реагируя на лавинообразный рост объемов генерируемых и обрабатываемых данных, требующих передачи по сети, на интенсивное развитие средств телекоммуникаций, появление и широкое внедрение перспективных информационно-коммуникационных технологий и сервисов. Их цели, ключевые задачи и основные функции непрерывно уточняются и трансформируются.

Статья организована следующим образом. В разделе 1 обобщен мировой опыт использования национальных сетей науки и образования для суперкомпьютерной инфраструктуры, в том числе рассмотрены крупнейшие проекты PRACE и XSEDE. Раздел 2 посвящен процессу интеграции ведомственных научно-образовательных сетей RASNet и RUNNet в национальную сеть науки и образования. В разделе 3 рассмотрены вопросы создания прикладной цифровой платформы, объединяющей в единую сеть вычислительные ресурсы территориально распределенных суперкомпьютерных центров коллективного пользования. В разделе 4 определены требования, предъявляемые к национальной сети науки и образования со стороны создаваемой прикладной цифровой платформы. В заключении кратко обобщены сформулированные требования и намечены направления дальнейших исследований.

1. Мировой опыт использования национальных сетей науки и образования для суперкомпьютерной инфраструктуры

К настоящему моменту национальные сети науки и образования существуют в большинстве стран мира, функционируют в качестве неотъемлемой части национальной ИКТ-инфраструктуры, как правило, координируются государственными органами управления наукой и образованием, представляют страну в международных проектах, при реализации которых интенсивно используются современные средства телекоммуникаций, развитые сетевые технологии и прикладные сервисы.

Примерами таких сетей являются DFN (Германия) [2], CANARIE (Канада) [3], Internet2 (США) [4], SURFnet (Нидерланды) [5], AARnet (Австралия) [6], CERNET (Китай) [7].

Для эффективной реализации международных проектов и взаимной кооперации создан ряд наднациональных объединений — консорциумов научно-образовательных сетей:

NORDUnet (Скандинавские страны) [8], GÉANT (Европа) [9], Asi@Connect/TEIN [10] и APAN (Азия) [11], RedClara (Южная Америка) [12], AfricaConnect [13].

Национальные сети науки и образования и их сообщества всегда использовались в качестве транспортной инфраструктуры для проектов объединения суперкомпьютерных ресурсов, например, проектов TeraGrid [14] и XSEDE [15] в США, DEISA [16] и PRACE [17] в Европе, NAREGI [18] в Японии.

Для таких проектов национальные сети науки и образования и их международные объединения развивают сервисы передачи данных со специальными требованями по качеству обслуживания: пропускной способности, уровню задержек и потерь пакетов, надежности работы и резервированию каналов связи. Ведутся разработки средств оперативного (в режиме онлайн) управления сетевыми ресурсами научных телекоммуникационных сетей: выделения и настройки каналов связи между научными организациями по требованию и расписанию.

Проект TeraGrid (2001–2011) [14] использовал выделенную волоконно-оптическую магистральную сеть 40 Гбит/с между Национальной лабораторией в Аргонне (Argonne National Laboratory), Калифорнийским технологическим институтом (California Institute of Technology), Национальным центром суперкомьютерных приложений (National Center for Supercomputing Applications) и Суперкомпьютерным центром в Сан-Диего (San Diego Supercomputer Center). Пользователи получали доступ к ресурсам проекта через национальные исследовательские сети: Internet2, Abilene backbone и National LambdaRail.

В проекте DEISA (2002–2011) [16] организации, предоставлявшие суперкомпьютерные ресурсы, подключались на скорости 10 Гбит/с к сети, которая была организована совместно общеевропейской магистральной научно-образовательной сетью GÉANT и национальными сетями науки и образования стран, участвовавших в проекте DEISA.

Для проекта PRACE — Partnership for Advanced Computing in Europe (2010–наст. время) — сетевой консорциум GÉANT построил новую высокоскоростную сеть на основе сервиса Многодоменной виртуальной частной сети (Multidomain Virtual Private Network, MD-VPN) [19].

Особенностью частной сети MD-VPN является организация наложенных сетей L3VPN (IPv4/IPv6), каналов L2VPN и многоточечных сетей L2VPN через несколько сетевых провайдеров (доменов) [20]. Конечным пользователям сервис предоставляется через так называемую точку демаркации услуг (Service Demarcation Point — SDP) на границе национальной сети науки и образования или даже региональной сети. На практике способ предоставления услуги конечным пользователям зависит от национальной сети науки и образования, обычно это L3VPN в форме IP-пакетов на третьем (сетевом) уровне и VLAN на порту или в пакете 802.1q для коммутируемых операторских сетей на основе протокола Ethernet или двухточечный L2VPN/многоточечный VPLS для сетей MPLS. В результате конечные пользователи могут работать в сетях IPv4/IPv6 или Ethernet так, как если бы их сети были связаны напрямую (промежуточные сети прозрачны для конечных пользователей).

Таким образом, суперкомпьютерные центры в разных странах могут работать так, как если бы они находились географически в одном месте; при этом их сети имеют равные уровни безопасности. Это позволяет избежать использования защитных экранов с контролем пакетов по их содержимому (deep packet inspection, DPI), как это потребовалось бы при использовании обычного протокола IP, увеличивает производительность сети и снижает за-

держки. Этот фактор очень важен для распределенных Grid-инфраструктур, облачных и высокопроизводительных вычислений.

Отметим, что служба MD-VPN также обеспечивает более высокий уровень конфиденциальности по сравнению с обычным VPN, так как потоки данных клиента MD-VPN изолированы от любого другого трафика: стандартного IP-трафика и трафика других клиентов MD-VPN.

Наиболее сложные требования к научно-образовательной сети предъявляет проект Extreme Science and Engineering Discovery Environment — XSEDE (2011–наст. время) [21]. В части обеспечения передачи данных требования включают:

- возможность передачи больших объемов данных, включая большие файлы и большое число файлов;
- информирование в реальном времени о состоянии сети и его изменениях в машиночитаемых форматах для взаимодействия с автоматизированными службами XSEDE;
- наличие механизмов обеспечения зарезервированной пропускной способности сети для отдельных пользователей или групп пользователей;
- наличие интегрированных со службами динамического распределения ресурсов сети механизмов управления для временного повышения скорости передачи данных для отдельных пользователей или групп пользователей;
- реализацию и поддержку служб и распределенных компонентов XSEDE и интерфейсов к ним непосредственно в телекоммуникационной сети.

В части сетевой безопасности требования включают:

- наличие общей системы обнаружения вторжений, которая сопоставляет информацию с различных датчиков на сетевом и системном уровне и способна отправлять уведомления в режиме реального времени сотрудникам службы безопасности XSEDE;
- наличие системы сканирования уязвимостей.

Кроме того, XSEDE предъявляет ряд требований по сетевой связности: подключение к телекоммуникационным сетям, поддерживаемым Национальным научным фондом США и правительственными агентствами DOE, DOD, NASA, NIH, зарубежным национальным научно-образовательными сетям, и тесное сотрудничество с ними в области развития сетевых технологий и сервисов.

Таким образом, суперкомпьютерные инфраструктуры предъявляют специальные требования к телекоммуникационной сети по передаче данных и наличию ряда дополнительных сервисов. Эти требования выходят далеко за рамки услуг коммерческих операторов связи и, как правило, могут быть удовлетворены только объединенными усилиями национальных научно-образовательных сетей.

2. Создание национальной компьютерной сети нового поколения на основе интеграции ведомственных научно-образовательных сетей

С начала 1990-х гг. в Российской Федерации при целевой поддержке федеральных органов управления образованием и наукой и иных государственных органов создавались и на протяжении долгих лет успешно эксплуатировались в интересах научно-образовательного сообщества отраслевые информационно-телекоммуникационные сети.

Российское научно-образовательное телекоммуникационное пространство исторически строилось в виде ряда взаимодействующих относительно независимых сетей, в значи-

тельной степени базирующихся на собственной канальной структуре и собственных же международных каналах. Некоторые из таких сетей изначально задумывались и проектировались как сети общего назначения, а другие — как специализированные предметно-ориентированные сети.

На конец 2018 года наиболее значимыми отраслевыми сетями сферы образования и науки являлись:

– Университетская сеть федерального масштаба RUNNet объединяет около 120 организаций высшего образования и науки из всех федеральных округов. Среди них — 70 ведущих университетов, в числе которых МГУ им М.В. Ломоносова, 12 университетов Программы 5/100 и 19 национальных исследовательских университетов.
– Система региональных сетей отделений РАН — RASNet. Наибольшее представительство пользователей RASNet сосредоточено в Московском регионе, где пользователями сети являются около 100 научных организаций.

В связи с разделением в 2018 году Минобрнауки России на два ведомства оператору сети RASNet (Межведомственный суперкомпьютерный центр РАН — МСЦ РАН) и оператору RUNNet (Центр реализации государственной образовательной политики и информационных технологий) было предложено инициировать мероприятия по объединению сетей на базе МСЦ РАН для создания национальной исследовательской компьютерной сети нового поколения (НИКС), которая будет играть роль национальной научно-образовательной сети России на международной арене. Концепция создания НИКС была одобрена на Заседании Совета Министерства науки и высшего образования Российской Федерации по информационно-телекоммуникационной инфраструктуре, информационной безопасности и суперкомпьютерным технологиям 24 апреля 2019 года.

МСЦ РАН обладает многолетним опытом управления глобальной ИКТ-инфраструктурой специализированного назначения, ее поддержки и эксплуатации, который включает управление сетью RASNet на федеральном уровне, обеспечение взаимодействия российских и зарубежных сетей науки и образования, представление страны в международных научных проектах, связанных с развитием научной телекоммуникационной и суперкомпьютерной инфраструктуры (GÉANT, DEISA, PRACE).

Передачу проекта RUNNet в МСЦ РАН и интеграцию сетей RASNet и RUNNet в национальную сеть науки и образования предполагается завершить до конца 2019 года.

Создание и развитие национальной исследовательской компьютерной сети нового поколения должно обеспечить функциональные возможности организациям науки и образования в следующих направлениях:

- сетевая связность и управление сетью:
  – удаленный доступ с обеспечением гарантированного качества сервиса и надежности для научно-образовательных организаций и исследовательских организаций промышленности к установкам класса «мегасайенс», уникальным научным установкам, центрам коллективного пользования научным оборудованием, суперкомпьютерным центрам, научным данным и результатам исследований (в т.ч. сверхбольшим объемам данных), ресурсам организаций науки и образования и сервисам с использованием собственной инфраструктуры и инфраструктуры мировых научно-образовательных сетей;
  – организация постоянной или по требованию/расписанию передачи данных со специальными условиями, в т.ч. выделение каналов связи и организация наложен-

ных сетей с заданным качеством обслуживания (пропускная способность, задержки, потери, надежность, резервирование), конфиденциальности и безопасности для совместных научных проектов, а также корпоративных сетей научно-образовательных организаций, объединяющих отделения и филиалы;
  - сквозной мониторинг пути передачи данных и виртуальных частных сетей, включая предоставление информации в машиночитаемых форматах для взаимодействия с автоматизированными службами;
  - интерфейс управления телекоммуникационными ресурсами и сервисами (заказ, выделение, мониторинг, учет для прикладных цифровых платформ, разрабатываемых организациями науки и высшего образования);
- безбарьерный доступ и безопасность:
  - обеспечение конфиденциальности передаваемых и обрабатываемых научных данных, организация доверенной среды передачи, обработки и хранения информации;
  - федеративная аутентификация и авторизация для безбарьерного доступа к российским и зарубежным научным ресурсам и данным;
  - защита персональных данных;
  - мобильность пользователей и безопасный доступ к научным данным и ресурсам из публичных сетей;
  - предоставление интерфейсов и инструментов федеративного доступа и безопасности для прикладных цифровых платформ, создаваемых и развиваемых организациями науки и образования.

Создаваемая национальная исследовательская компьютерная сеть нового поколения должна интегрироваться в мировое сообщество научно-образовательных сетей в роли российской национальной сети науки и образования, что будет способствовать переносу и распространению в России опыта зарубежных научно-образовательных сетей в развитии телекоммуникационных сервисов и инструментов для поддержки суперкомпьютерных инфраструктур.

3. Проект создания распределенной сети суперкомпьютерных центров коллективного пользования

В настоящее время в МСЦ РАН ведутся работы по созданию прикладной цифровой платформы (ПЦП), объединяющей в единую сеть вычислительные ресурсы территориально распределенных суперкомпьютерных центров коллективного пользования [22] в интересах организаций науки, высшего образования и инновационной деятельности Российской Федерации. Целями подобного объединения являются оптимизация распределения вычислительной нагрузки, упрощение и повышение качества доступа пользователей к высокопроизводительным вычислительным ресурсам, консолидация функций мониторинга и управления распределенными суперкомпьютерными ресурсами.

Для создания платформы необходимы исследования и разработка новых методов и алгоритмов, касающихся всех аспектов функционирования распределенной сети СКЦ: организации единого доступа к пулу объединенных ресурсов, распределения пользовательских заданий по вычислительным ресурсам, мониторинга и прогнозирования состояния вычислительной среды, организации централизованного управления данными, моделирования и анализа работы всей системы.

Создание ПЦП требует разработки и создания:

- децентрализованной автоматизированной системы управления заданиями и ресурсами, которая будет поддерживать глобальную очередь пользовательских заданий и обеспечивать за счет этого оперативное перераспределение вычислительной нагрузки в сети СКЦ;
- единой системы мониторинга, которая позволит оперативно получать информацию о текущих состоянии и загруженности суперкомпьютерных ресурсов распределенной сети;
- единой системы доступа на основе удостоверяющей федерации суперкомпьютерных центров, которая обеспечит пользователей технологией единого входа (Single Sign-On, SSO) на суперкомпьютерные ресурсы распределенной сети, причем авторизоваться в сети можно будет с использованием только личной учетной записи в своей организации;
- общей распределенной системы хранения данных, обеспечивающей единое файловое пространство для всех СКЦ сети.

Создание системы управления сетью СКЦ является одной из ключевых задач проекта. Система управления имеет децентрализованный характер и базируется на схеме асинхронного взаимодействия коллектива равноправных диспетчеров [23]. Взаимодействие диспетчеров осуществляется через единую информационную подсистему, в которой формируется и хранится глобальная очередь пользовательских заданий [24]. Основой информационной подсистемы является распределенная документо-ориентированная СУБД Elasticsearch, причем такой подход позволяет сочетать достоинства централизованной (логическая простота взаимодействия) и децентрализованной (отказоустойчивость и надежность хранения) схем управления.

Несмотря на высокую востребованность отечественных суперкомпьютерных систем, их средняя загрузка часто не превышает 70–80%. При утилизации ресурсов 90% и выше (как, например, в МСЦ РАН или НИВЦ МГУ им. М.В. Ломоносова), загруженность того или иного СКЦ будет определяться средним временем нахождения задания в очереди. Предлагаемая асинхронная децентрализованная схема управления позволит через глобальную очередь организовать оперативное перераспределение заданий в сети СКЦ из более загруженного центра в менее загруженный. За счет этого удастся сократить время пребывания задания в очереди и таким образом снизить время ожидания пользователем результатов расчетов.

Существенной проблемой при организации глобальной очереди заданий является обеспечение бинарной переносимости исполняемых модулей пользовательских приложений [25]. Для решения этой проблемы в МСЦ РАН ведутся интенсивные исследования в области контейнерной виртуализации. Найденные методы и способы [26] представления пользовательских заданий в виде контейнеров могут быть применены при организации перераспределения заданий глобальной очереди в сети СКЦ. Развитие технологий виртуализации в настоящее время позволяет говорить о программно-определяемой инфраструктуре вычислительных центров, в которой виртуализованные ключевые подсистемы (вычислительная, сетевая и хранения данных) предоставляются пользователям в виде сервисов с заданным качеством. Показано [27], что реализация такой инфраструктуры позволяет обеспечить возможность каждому пользователю продуктивно решать свои задачи за приемлемое время с приемлемым уровнем затрат.

Немаловажным вопросом при построении системы управления является разработка алгоритмов глобального планирования заданий. Проведенные исследования [28] показали, что в случае применения децентрализованной схемы управления с равноправными диспетчерами хорошо работают аукционные методы планирования. Для случая абсолютных приоритетов заданий наибольшую эффективность демонстрирует алгоритм планирования, основанный на методе английского аукциона [28].

Децентрализованная схема управления позволяет поддерживать независимость отдельных СКЦ из состава сети, что соответствует существующей организационной структуре: центрами владеют и управляют разные научные организации в разных ведомствах (Минобрнауки России, МГУ им. М.В. Ломоносова, НИЦ «Курчатовский институт»). Негативным следствием децентрализации управления является необходимость регистрации каждого пользователя в каждом СКЦ, где ведется независимая собственная база (реестр) пользователей. В каждом СКЦ возникает непрофильная для научной организации проблема обработки и защиты персональных данных пользователей. В подавляющем большинстве случаев персональные данные пользователя требуются только для экстренной связи с ним, а также для формирования агрегированной обезличенной статистики (сколько молодых исследователей используют суперкомпьютерные ресурсы, каков процент пользователей с ученой степенью и т.п.). Подобного вида статистика без юридических и технических препятствий может быть запрошена суперкомпьютерным центром у организации — работодателя пользователя.

Отметим, что распределение квот вычислительных ресурсов в СКЦ удобно осуществлять по иерархической схеме: вначале квоты выделяются организациям и научным проектам, а далее — группам пользователей и пользователям в рамках определенного проекта. Руководители проектов и организаций заинтересованы в возможности самостоятельного перераспределения квот на ресурсы между своими проектами и пользователями.

Приемлемым решением видится создание распределенной системы аутентификации, авторизации и учета, в которой пользователь будет иметь единственную учетную запись, а институты могли бы гибко управлять выделяемыми или приобретаемыми суперкомпьютерными ресурсами. Подходящей основой для такой системы может стать удостоверяющая федерация [29] из суперкомпьютерных центров/институтов, доверяющих друг другу аутентификацию пользователя. Разрабатываемое в МСЦ РАН решение по построению удостоверяющей федерации сети СКЦ подробно рассмотрено в [30].

Единая система мониторинга предназначена для оперативного сбора информации о состоянии и загруженности сети СКЦ, ее хранения и отображения. Среди основных функций системы мониторинга следует отметить отслеживание состава сети СКЦ, характеристик входящих в состав суперкомпьютерных ресурсов; отображение текущей загруженности суперкомпьютерных ресурсов; мониторинг содержания выполняемой вычислительной работы (какие пользователи, где, и какие задания выполняют); отслеживание и отображение объемов израсходованных ресурсов, текущего состояния бюджета, выделенных квот ресурсов, настроек приоритетов, графиков изменения этих характеристик по всем научным проектам; мониторинг состояния вычислительных узлов суперкомпьютеров и каналов связи.

Проектом создаваемой сети СКЦ предусматривается выделение отдельного уровня отказоустойчивого хранения и глобального доступа к данным. На этом уровне пользователям предоставляется общее пространство хранения данных, доступное из любого СКЦ. Для организации общего пространства часть ресурсов систем хранения данных СКЦ использу-

ется для развертывания программно-определяемого хранилища данных — узла облачной системы хранения. На объединяемых узлах облачной системы хранения реализуется единое пространство имен файлов [31]. Доступ пользователей к данным организуется через шлюз облачной системы хранения [32], через который реализуются механизмы кэширования данных и политики автоматического перемещения «остывших» данных в облачную систему хранения. Надежность хранения данных в облачной системе хранения обеспечивается либо репликацией данных, либо использованием алгоритмов рассредоточения информации [33] между узлами облачной системы хранения.

4. Требования к телекоммуникационной инфраструктуре НИКС и сервисам на ее основе со стороны распределенной сети суперкомпьютерных центров коллективного пользования

Процесс выполнения высокопроизводительных вычислений можно разделить на три основных этапа:

– этап подготовки исходных данных;
– этап расчетов;
– этап анализа результатов и их визуализации.

На каждом из этапов задействуется свой вид вычислительной техники. Для подготовки данных требуется многопроцессорный сервер с большим объемом оперативной памяти (в настоящее время — несколько терабайт). Особая важность этапа подготовки (предобработки) данных заключается в том, что от точности исходных расчетных сеток зачастую зависит скорость и точность высокопроизводительных расчетов.

На этапе расчетов задействуются суперкомпьютерные ресурсы. Как правило, этот этап занимает наибольшее количество времени и вычислительных ресурсов.

На этапе анализа результатов и визуализации (постобработки) требуются рабочие станции, оснащенные развитой графической подсистемой.

В распределенной сети СКЦ серверы предобработки, суперкомпьютеры и станции постобработки могут принадлежать различным организациям и находиться на значительном территориальном удалении друг от друга. Поскольку связующим звеном между названными этапами высокопроизводительных вычислений являются пользовательские данные, для их автоматического и прозрачного для пользователя перемещения от серверов предобработки к суперкомпьютерам, от суперкомпьютеров к станциям постобработки, необходима единая для всей сети распределенная файловая система.

Отсутствие единого файлового пространства усложнит процесс управления вычислениями и распределения вычислительной нагрузки между суперкомпьютерами, а главное, усложнит работу пользователей.

Организация и функционирование такой файловой системы требует наличия надежной сетевой инфраструктуры, обеспечивающей гарантированные пропускную способность, низкую латентность и быстрое время отклика.

Рассмотрим минимальные требования к гарантированной пропускной способности телекоммуникационной сети для обеспечения работоспособности сети СКЦ. Размер исходных данных для расчетов и их результатов для одного задания в среднем составляет 100 Гбайт. Статистика МСЦ РАН показывает, что за год суперкомпьютеры центра выполняют около 100 тыс. заданий, т.е. средняя интенсивность входного потока — 1 задание в 5 минут.

Согласно результатам моделирования [28] перераспределению (перемещению между СКЦ) подвергается около 50% всех заданий, т.е. 1 задание в 10 минут.

Для перемещения в течение этого времени задания размером 100 Гбайт с учетом накладных расходов (заголовки кадров Ethernet и служебная информация) потребуется сеть передачи данных, обеспечивающая постоянную скорость передачи данных минимальной пропускной способностью 1,4 Гбит/с. Учитывая неравномерность передачи данных и особенности передачи данных на большие дистанции [34], только для передачи служебного трафика, вызванного перераспределеним задач, с уровнем задержек, не вызывающим перебои функционирования, потребуется пропускная способность на порядок больше — 10–15 Гбит/с. С учетом трафика удаленно работающих пользователей (загрузки и выгрузки данных в системы хранения сети СКЦ из научных организаций) требуемая пропускная способность возрастает до 15–20 Гбит/с.

В ближайшие три–пять лет следует ожидать рост производительности суперкомпьютерных систем в 2–3 раза, что приведет к росту детализации расчетных сеток и, как результат, к росту объема исходных данных и результатов в 2–3 раза, а следовательно, к росту требований к пропускной способности телекоммуникационной сети, необходимой для нормального функционирования сети СКЦ до 40 Гбит/с и выше.

Требования к пропускной способности телекоммуникационной сети можно снизить, если возможно предсказать гарантированное время доставки данных пользователя, и учесть его в процессе планирования распределения задач. Это возможно, если телекоммуникационная сеть будет иметь механизм выделения гарантированной полосы пропускания по требованию или по расписанию и предоставит планировщикам заданий сети СКЦ прикладной программный интерфейс к этому механизму.

Дополнительные требования по функциональности возникают при использовании контейнерного представляения заданий. Например, если исполняемое в интерактивном режиме задание получит дополнительные ресурсы (вычислительные узлы) на удаленной машине, связь с ними может быть организована через виртуальную частную сеть, которую необходимо создать средствами телекоммуникационной сети. Для этого телекоммуникационная сеть должна предоставить планировщику заданий сети СКЦ сответствующий прикладной программный интерфейс управления виртуальными частными сетями.

Упомянутые выше сервисы автоматизированного выделения сетевых ресурсов с использованием прикладных программных интерфейсов разрабатываются и уже предоставляются некоторыми зарубежными сетями науки и образования. Такие сервисы планируются и в национальной исследовательской компьютерной сети нового поколения.

Для функционирования сети СКЦ необходим непрерывный мониторинг как сетевой связности, так и характеристик обеспечиваемого качества связи. Национальные научно-образовательные сети обеспечивают такой мониторинг конечному пользователю на основе инструмента PerfSONAR [35]. Внедрение этого инструмента в НИКС является одной из приоритетных задач.

Для перераспределения задания пользователя с одного вычислительного ресурса сети СКЦ на другой необходимо поддерживать соответствие между учетными записями пользователя на этих ресурсах. Эта задача значительно упрощается при использовании единой учетной записи на всех ресурсах. Одним из способов реализации единой учетной записи является федеративная авторизация и аутентификация, при которой служба авторизации ресурса перенаправляет пользователя для аутентификации в организацию, в которой он

работает или учится. Сервисы федеративной аутентификации и авторизации, интегрированные в международные удостоверяющие федерации научно-образовательных сетей, уже поддерживаются в нашей стране сетями RASNet и RUNNet, а в рамках создаваемой на основе этих сетей НИКС планируется их дальнейшее развитие.

Заключение

Для полной реализации функциональности и обеспечения эффективности распределенной сети СКЦ необходима телекоммуникационная сеть со следующими требованиями:

— пропускная способность до 15–20 Гбит/с (с наращиванием в ближайшие 3–5 лет до 40 Гбит/с);
— автоматизированное выделение гарантированной полосы пропускания по требованию и по расписанию с использованием прикладного программного интерфейса;
— автоматизированное выделение виртуальных частных сетей по требованию и по расписанию с использованием прикладного программного интерфейса;
— средства сквозного мониторинга пути передачи данных с прикладным программным интерфейсом;
— поддержка федеративной аутентификации и авторизации.

В создаваемой национальной исследовательской компьютерной сети нового поколения планируется развитие функциональных возможностей, обеспечивающих выполение перечисленных выше требований, с учетом имеющегося опыта зарубежных национальных научно-образовательных сетей и интеграции российской национальной сети в мировое сообщество научно-образовательных сетей.

Публикация выполнена в МСЦ РАН в рамках государственного задания по проведению фундаментальных научных исследований.

Литература

1. Фортов В.Е., Савин Г.И., Левин В.К., Забродин А.В., Шабанов Б.М. Создание и применение системы высокопроизводительных вычислений на базе высокоскоростных сетевых технологий // Информационные технологии и вычислительные системы. 2002. № 1. С. 3.
2. Deutschen Forschungsnetz. URL: `https://www.dfn.de/` (дата обращения: 21.08.2019).
3. CANARIE. URL: `https://www.canarie.ca/` (дата обращения: 21.08.2019).
4. Internet2. URL: `https://www.internet2.edu/` (дата обращения: 21.08.2019).
5. SURFnet. URL: `https://www.surf.nl/en` (дата обращения: 21.08.2019).
6. AARNET. URL: `https://www.aarnet.edu.au/` (дата обращения: 21.08.2019).
7. China Educational and Research Network. URL: `http://www.edu.cn/english/` (дата обращения: 21.08.2019).
8. NORDUnet. Nordic gateway for Research and Education. URL: `https://www.nordu.net/` (дата обращения: 21.08.2019).
9. GÉANT. URL: `https://www.geant.org/` (дата обращения: 21.08.2019).
10. Asi@Connect. URL: `http://www.tein.asia` (дата обращения: 21.08.2019).
11. Asia Pacific Advanced Network. URL: `https://apan.net/` (дата обращения: 21.08.2019).

12. RedCLARA. Latin American Cooperation of Advanced Networks. URL: `https://www.redclara.net/` (дата обращения: 21.08.2019).

13. AfricaConnect2. URL: `https://www.africaconnect2.net/` (дата обращения: 21.08.2019).

14. Catlett C. The philosophy of TeraGrid: building an open, extensible, distributed TeraScale facility. Cluster Computing and the Grid 2nd IEEE/ACM International Symposium CCGRID2002, 2002. DOI: 10.1109/CCGRID.2002.1017101.

15. XSEDE — The Extreme Science and Engineering Discovery Environment. URL: `https://www.xsede.org/` (дата обращения: 21.08.2019).

16. Bassini S., Cavazonni C., Gheller C. European actions for High-Performance Computing: PRACE, DEISA and HPC-Europa. Il Nuovo Cimento C. 2009. Vol. 32. P. 93–97.

17. PRACE — Partnetship for Advanced Computing in Europe. URL: `http://www.prace-ri.eu/` (дата обращения: 21.08.2019).

18. Matsuoka S., Shimojo S., Aoyagi M., Sekiguchi S., Usami H., Miura K. Japanese Computational Grid Research Project: NAREGI. Proceedings of the IEEE. 2005. Vol. 93, no. 3. P. 522–533. DOI: 10.1109/JPROC.2004.842748.

19. PRACE: Europe's supercomputing infrastructure relies on GÉANT. URL: `https://impact.geant.org/portfolio/prace/` (дата обращения: 21.08.2019).

20. MD-VPN Product Description. URL: `https://wiki.geant.org/display/PLMTES/MD-VPN+Product+Description` (дата обращения: 21.08.2019).

21. XSEDE System Requirements Specification v3.1. URL: `http://hdl.handle.net/2142/45102` (дата обращения: 21.08.2019).

22. Шабанов Б.М., Овсянников А.П., Баранов А.В., Лещев С.А., Долгов Б.В., Дербышев Д.Ю. Проект распределенной сети суперкомпьютерных центров коллективного пользования // Программные системы: теория и приложения. 2017. № 4(35). С. 245–262. DOI: 10.25209/2079-3316-2017-8-4-245-262.

23. Шабанов Б.М., Телегин П.Н., Овсянников А.П., Баранов А.В., Тихомиров А.И., Ляховец Д.С. Система управления заданиями распределенной сети суперкомпьютерных центров коллективного пользования // Труды научно-исследовательского института системных исследований Российской академии наук. 2018. Т. 8, № 6. С. 65–73. DOI: 10.25682/NIISI.2018.6.0009.

24. Баранов А.В., Тихомиров А.И. Методы и средства организации глобальной очереди заданий в территориально распределенной вычислительной системе // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2017. Т. 6, № 4. С. 28–42. DOI: 10.14529/cmse170403.

25. Шабанов Б.М., Телегин П.Н., Баранов А.В., Семенов Д.В., Чуваев А.В. Динамический конфигуратор виртуальной распределенной вычислительной среды // Программные продукты, системы и алгоритмы. 2017. № 4. DOI: 10.15827/2311-6749.25.272.

26. Baranov A.V., Savin G.I., Shabanov B.M. *et al.* Methods of Jobs Containerization for Supercomputer Workload Managers // Lobachevskii Journal of Mathematics. 2019. Vol. 40, no. 5. P. 525–534. DOI: 10.1134/S1995080219050020.

27. Шабанов Б.М., Самоваров О.И. Принципы построения межведомственного центра коллективного пользования общего назначения в модели программно-определяемого ЦОД

// Труды Института системного программирования РАН. 2018. Т. 30, № 6. С. 7–24. DOI: 10.15514/ISPRAS-2018-30(6)-1.

28. Baranov A., Telegin P., Tikhomirov A. Comparison of Auction Methods for Job Scheduling with Absolute Priorities // In: Malyshkin V. (eds) Parallel Computing Technologies (PaCT 2017). Lecture Notes in Computer Science. 2017. Vol. 10421. P. 387–395. DOI: 10.1007/978-3-319-62932-2_37.

29. Овсянников А.П., Савин Г.И., Шабанов Б.М. Удостоверяющие федерации научно-образовательных сетей // Программные продукты и системы. 2012. № 4. С. 3–7.

30. Баранов А.В., Овсянников А.П., Шабанов Б.М. Федеративная аутентификация в распределенной инфраструктуре суперкомпьютерных центров // Труды научно-исследовательского института системных исследований Российской академии наук. 2018. Т. 8, № 6. С. 79–83. DOI: 10.25682/NIISI.2018.6.0011.

31. Koulouzis S., Belloum A., Bubak M., Lamata P., Nolte D., Vasyunin D., de Laat C. Distributed Data Management Service for VPH Applications // IEEE Internet Computing. 2016. Vol. 20, no. 2. P. 34–41. DOI: 10.1109/MIC.2015.71.

32. Kapadia A., Varma S., Rajana K. Implementing Cloud Storage with OpenStack Swift. Packt Publishing, 2014. 105 p.

33. Джонс М. Анатомия облачной инфраструктуры хранения данных. Модели, функции и внутренние детали. 2012. URL: `https://www.ibm.com/developerworks/ru/library/cl-cloudstorage/cl-cloudstorage-pdf.pdf` (дата обращения: 28.08.2019).

34. Баранов А.В., Вершинин Д.В., Дербышев Д.Ю., Долгов Б.В., Лещев С.А., Овсянников А.П., Шабанов Б.М. Об эффективности использования канала связи между территориально удаленными суперкомпьютерными центрами // Труды научно-исследовательского института системных исследований Российской академии наук. 2017. Т. 7, № 4. С. 137–142.

35. Hanemann A. *et al.* PerfSONAR: A Service Oriented Architecture for Multi-domain Network Monitoring // Lecture Notes in Computer Science. 2005. Vol. 3826. P. 241–254. DOI: 10.1007/11596141_19.

Савин Геннадий Иванович, академик РАН, д.ф.-м.н., профессор, научный руководитель Межведомственного суперкомпьютерного центра РАН — филиала ФГУ ФНЦ НИИСИ РАН (Москва, Российская Федерация)

Шабанов Борис Михайлович, к.т.н., доцент, директор Межведомственного суперкомпьютерного центра РАН — филиала ФГУ ФНЦ НИИСИ РАН (Москва, Российская Федерация)

Баранов Антон Викторович, к.т.н., доцент, в.н.с. Межведомственного суперкомпьютерного центра РАН — филиала ФГУ ФНЦ НИИСИ РАН (Москва, Российская Федерация)

Овсянников Алексей Павлович, в.н.с. Межведомственного суперкомпьютерного центра РАН — филиала ФГУ ФНЦ НИИСИ РАН (Москва, Российская Федерация)

Гончар Андрей Андреевич, в.н.с. Межведомственного суперкомпьютерного центра РАН — филиала ФГУ ФНЦ НИИСИ РАН (Москва, Российская Федерация)

DOI: 10.14529/cmse200102

ON THE USE OF FEDERAL SCIENTIFIC TELECOMMUNICATION INFRASTRUCTURE FOR HIGH PERFORMANCE COMPUTING

© 2020 G.I. Savin, B.M. Shabanov, A.V. Baranov, A.P. Ovsyannikov, A.A. Gonchar

Joint SuperComputer Center of the Russian Academy of Sciences — Branch of Federal State Institution "Scientific Research Institute for System Analysis of the Russian Academy of Sciences" (Leninsky Prospect 32a, Moscow, 119334 Russia)

E-mail: savin@jscc.ru, shabanov@jscc.ru, antbar@mail.ru, ovsiannikov@jscc.ru, andrey.gonchar@jscc.ru

Received: 09.01.2020

The article is devoted to the prospects for the development of scientific telecommunications infrastructure based on the new generation national research computer network (NRCN), formed by the integration of departmental scientific and educational networks RUNNet and RASNet. The new network′ capabilities for combining supercomputer resources and providing barrier-free access to them are shown. Based on the generalized world experience, it has been shown that supercomputer infrastructures have special requirements for a telecommunication network for data transmission and the presence of a number of additional services. These requirements go far beyond the services of commercial telecom providers and, as a rule, can only be satisfied by the combined efforts of national scientific and educational networks.

The key elements of the federal telecommunications infrastructure necessary for combining high-performance computing resources are considered: high-performance communication channels with a specified quality of service, their automatic allocation on demand and on schedule, trusted network environment, federated authentication and authorization, reliability and security, end-to-end monitoring of the data transmission path between end users. Based on the analysis of the life cycle of the supercomputer job migrating at the distributed network, the requirements for the NRCN telecommunications infrastructure and services based on it are formulated.

Keywords: national science and education network, supercomputer center, shared research facilities, distributed computing, telecommunications infrastructure.

FOR CITATION

Savin G.I., Shabanov B.M., Baranov A.V., Ovsyannikov A.P., Gonchar A.A. On the Use of Federal Scientific Telecommunication Infrastructure for High Performance Computing. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2020. Vol. 9, no. 1. P. 20–35. (in Russian) DOI: 10.14529/cmse200102.

References

1. Fortov V.E., Savin G.I., Levin V.K., Zabrodin A.V., Shabanov B.M. Creation and application of a high-performance computing system based on high-speed network technologies. Journal of Information Technologies and Computing. 2002. no. 1. P. 3. (in Russian)
2. Deutschen Forschungsnetz. Available at: `https://www.dfn.de/` (accessed: 21.08.2019).
3. CANARIE. Available at: `https://www.canarie.ca/` (accessed: 21.08.2019).
4. Internet2. Available at: `https://www.internet2.edu/` (accessed: 21.08.2019).

5. SURFnet. Available at: https://www.surf.nl/en (accessed: 21.08.2019).

6. AARNET. Available at: https://www.aarnet.edu.au/ (accessed: 21.08.2019).

7. China Educational and Research Network. Available at: http://www.edu.cn/english/ (accessed: 21.08.2019).

8. NORDUnet. Nordic gateway for Research and Education. Available at: https://www.nordu.net/ (accessed: 21.08.2019).

9. GÉANT. Available at: https://www.geant.org/ (accessed: 21.08.2019).

10. Asi@Connect. Available at: http://www.tein.asia (accessed: 21.08.2019).

11. Asia Pacific Advanced Network. Available at: https://apan.net/ (accessed: 21.08.2019).

12. RedCLARA. Latin American Cooperation of Advanced Networks. Available at: https://www.redclara.net/ (accessed: 21.08.2019).

13. AfricaConnect2. Available at: https://www.africaconnect2.net/ (accessed: 21.08.2019).

14. Catlett C. The philosophy of TeraGrid: building an open, extensible, distributed TeraScale facility. Cluster Computing and the Grid 2nd IEEE/ACM International Symposium (CCGRID 2002). 2002. DOI: 10.1109/CCGRID.2002.1017101.

15. XSEDE — The Extreme Science and Engineering Discovery Environment. Available at: https://www.xsede.org/ (accessed: 21.08.2019).

16. Bassini S., Cavazonni C., Gheller C. European actions for High-Performance Computing: PRACE, DEISA and HPC-Europa. Il Nuovo Cimento C. 2009. Vol. 32. P. 93–97.

17. PRACE — Partnetship for Advanced Computing in Europe. Available at: http://www.prace-ri.eu/ (accessed: 21.08.2019).

18. Matsuoka S., Shimojo S., Aoyagi M., Sekiguchi S., Usami H., Miura K. Japanese Computational Grid Research Project: NAREGI. Proceedings of the IEEE. 2005. Vol. 93, no. 3. P. 522–533. DOI: 10.1109/JPROC.2004.842748.

19. PRACE: Europe's supercomputing infrastructure relies on GÉANT. Available at: https://impact.geant.org/portfolio/prace/ (accessed: 21.08.2019).

20. MD-VPN Product Description. Available at: https://wiki.geant.org/display/PLMTES/MD-VPN+Product+Description (accessed: 21.08.2019).

21. XSEDE System Requirements Specification v3.1. Available at: http://hdl.handle.net/2142/45102 (accessed: 21.08.2019).

22. Shabanov B., Ovsiannikov A., Baranov A., Leshchev S., Dolgov B., Derbyshev D. The distributed network of the supercomputer centers for collaborative research. Program systems: Theory and applications. 2017. no. 8:4(35). P. 245–262. (in Russian) DOI: 10.25209/2079-3316-2017-8-4-245-262.

23. Shabanov B.M., Telegin P.N., Ovsyannikov A.P., Baranov A.V., Tikhomirov A.I., Lyakhovets D.S. The Jobs Management System for the Distributed Network of the Supercomputer Centers. The Proceeding of the Scientific Research Institute for System Analysis of the Russian Academy of Sciences. 2018. Vol. 8, no. 6. P. 65–73. (in Russian) DOI: 10.25682/NIISI.2018.6.0009.

24. Baranov A.V., Tikhomirov A.I. Methods and Tools for Organizing the Global Job Queue in the Geographically Distributed Computing System. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2017. Vol. 6, no. 4. P. 28–42. (in Russian) DOI: 10.14529/cmse170403.

25. Shabanov B.M., Telegin P.N., Baranov A.V., Semenov D.V., Chuvaev A.V. Dynamic Configurator for Virtual Distributed Computing Environment. Software Journal: Theory and Applications. 2017. no. 4. (in Russian) DOI: 10.15827/2311-6749.25.272.

26. Baranov A.V., Savin G.I., Shabanov B.M. *et al.* Methods of Jobs Containerization for Supercomputer Workload Managers. Lobachevskii Journal of Mathematics. 2019. Vol. 40, no. 5. P. 525–534. DOI: 10.1134/S1995080219050020.

27. Shabanov B.M., Samovarov O.I. Building the Software Defined Data Center. Proceedings of the Institute for System Programming. 2018. Vol. 30, no. 6. P. 7–24. (in Russian) DOI: 10.15514/ISPRAS-2018-30(6)-1.

28. Baranov A., Telegin P., Tikhomirov A. Comparison of Auction Methods for Job Scheduling with Absolute Priorities. In: Malyshkin V. (eds) Parallel Computing Technologies (PaCT 2017). Lecture Notes in Computer Science. 2017. Vol. 10421. P. 387–395. DOI: 10.1007/978-3-319-62932-2_37.

29. Ovsyannikov A.P., Savin G.I., Shabanov B.M. Identity federation of the research and educational networks. Software & Systems. 2012. no. 4. P. 3–7. (in Russian)

30. Baranov A.V., Shabanov B.M., Ovsyannikov A.P. Federative Identity for the Distributed Infrastructure of the Supercomputer Centers. The Proceeding of the Scientific Research Institute for System Analysis of the Russian Academy of Sciences. 2018. Vol. 8, no. 6. P. 79–83. (in Russian) DOI: 10.25682/NIISI.2018.6.0011.

31. Koulouzis S., Belloum A., Bubak M., Lamata P., Nolte D., Vasyunin D., de Laat C. Distributed Data Management Service for VPH Applications. IEEE Internet Computing. 2016. Vol. 20, no. 2. P. 34–41. DOI: 10.1109/MIC.2015.71.

32. Kapadia A., Varma S., Rajana K. Implementing Cloud Storage with OpenStack Swift. Packt Publishing, 2014. 105 p.

33. Jones M. Anatomy of a cloud storage infrastructure. Models, features, and internals. 2010. Available at: `https://developer.ibm.com/articles/cl-cloudstorage/` (accessed: 21.08.2019).

34. Baranov A.V., Derbyshev D.Yu., Dolgov B.V., Leshchev S.A., Ovsyannikov A.P., Shabanov B.M., Vershinin D.V. Effective usage of the link between geographically distributed supercomputer centers. The Proceeding of the Scientific Research Institute for System Analysis of the Russian Academy of Sciences. 2017. Vol. 7, no. 4. P. 137–142. (in Russian)

35. Hanemann A. *et al.* PerfSONAR: A Service Oriented Architecture for Multi-domain Network Monitoring. Lecture Notes in Computer Science. 2005. Vol. 3826. P. 241–254. DOI: 10.1007/11596141_19.

УДК 004.051 DOI: 10.14529/cmse200103

НЕЙРОСЕТЕВОЙ МЕТОД РЕШЕНИЯ ЗАДАЧИ МЭППИНГА ПАРАЛЛЕЛЬНЫХ ПРИЛОЖЕНИЙ*

© 2020 Н.Н. Попова, М.В. Козлов, М.В. Шубин

Московский государственный университет имени М.В. Ломоносова
(119991 Москва, ул. Ленинские горы, д. 1)
E-mail: popova@cs.msu.su, rat.taurus@gmail.com, mihshub@gmail.com
Поступила в редакцию: 12.11.2019

Статья посвящена проблеме повышения эффективности параллельных приложений. В статье предлагается подход к решению проблемы, основанный на сокращении накладных расходов, связанных с передачей данных между процессами параллельной программы во время ее выполнения на высокопроизводительной вычислительной системе. С ростом числа процессорных узлов расходы на передачу сообщений между узлами оказывают все большее влияние на производительность параллельных приложений. В связи с этим становится особо актуальной задача размещения процессов параллельной программы по вычислительным узлам суперкомпьютера, известная, как задача мэппинга. В работе предлагается новый подход к решению задачи мэппинга. Ключевой особенностью подхода является выбор коммуникационного шаблона путем фазового анализа приложения и использование сверточной нейронной сети для быстрого выбора подходящего алгоритма мэппинга, исходя из построенного коммуникационного шаблона.

Для построения коммуникационных шаблонов проводится анализ поведения приложения с точки зрения передачи сообщений точка-точка между процессами параллельной программы. Временная шкала событий передачи сообщений разбивается на равные промежутки, для каждого из которых строится коммуникационных шаблон. К построенным шаблонам применяется двумерное вейвлет-преобразование Хаара для выделения признаков. Затем проводится кластеризация признаков и построение фаз во временной шкале работы приложения. Для каждой фазы строится коммуникационный шаблон, соответствующий этой фазе.

Выбор подходящего алгоритма мэппинга проводится с помощью сверточной нейронной сети. Использование нейронной сети предполагает знание о свойствах коммуникационного поведения различных типов приложений и подходящих для этих типов алгоритмов мэппинга. Эти знания должны быть представлены в виде набора классов коммуникационных шаблонов (матриц) с известным для каждого класса наилучшим алгоритмом мэппинга. Нейронная сеть обучается на данном наборе классов. Обученная сеть решает задачу классификации входного коммуникационного шаблона, выбирая наиболее подходящий алгоритм мэппинга для данного параллельного приложения.

В статье представлена реализация отдельных этапов метода, и продемонстрирована их работа на тестовых примерах.

Ключевые слова: суперкомпъютер, мэппинг, коммуникационный шаблон, сверточная нейронная сеть.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Попова Н.Н., Козлов М.В., Шубин М.В. Нейросетевой метод решения задачи мэппинга параллельных приложений // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 1. С. 36–49. DOI: 10.14529/cmse200103.

Введение

Одним из путей повышения эффективности параллельных приложений является оптимизация размещения процессов параллельной программы по узлам и ядрам целевой вычислительной системы, на которой выполняется приложение. Задача об оптимальном размещении, получившая название мэппинг параллельных приложений, активно исследуется. Решение этой задачи приобретает особую ценность для высокопроизводительных систем

*Статья рекомендована к публикации программным комитетом Международной конференции «Суперкомпьютерные дни в России — 2019».

(HPC-систем) рекордной производительности [1]. В основе большинства предлагаемых подходов к решению задачи мэппинга лежит модель взаимодействия параллельных процессов, представленная в виде коммуникационных матриц, элементами которых являются численные характеристики сообщений, передаваемых между каждой из пар взаимодействующих процессов. В статье предлагается уточнение этой модели. Уточнение проводится в двух направлениях: альтернативного подхода к построению коммуникационной матрицы и нового подхода к определению мэппинга, основанного на использовании сверточных нейронных сетей. Оба направления предлагаемого подхода основаны на использовании алгоритмов машинного обучения. Хотя каждое из направлений является самодостаточным, они могут использоваться в различных задачах, связанных с анализом поведения и настройкой эффективности параллельных программ.

Задача мэппинга является NP-полной [2]. Для ее решения широко используются разнообразные эвристики. Однако, далеко не каждый алгоритм мэппинга уменьшает время выполнения программы, а некоторые даже его увеличивают. В качестве подтверждения этого можно привести примеры, представленные в статье [3]. В данной работе показано, что в зависимости от используемого алгоритма мэппинга, время выполнения приложения может как уменьшаться по сравнению с принятым по умолчанию стандартным мэппингом, так и увеличиваться. На подбор «правильного» алгоритма для данной задачи может уйти много времени и затрачены значительные вычислительные ресурсы. Вследствие этого определение подходящего алгоритма мэппинга для данной программы является еще одним способом оптимизации параллельных программ.

Основное содержание этапов предлагаемого подхода заключается в следующем. Коммуникационная матрица параллельного приложения рассматривается как изображение. На основе анализа динамики изменения коммуникационных взаимодействий выбирается коммуникационная матрица, отвечающая наиболее характерному паттерну взаимодействия процессов. Для построения мэппинга формируется сверточная нейронная сеть, которая относит коммуникационную матрицу, рассматриваемую как изображение, к одному из заданных классов. Для каждого из заданных классов заранее определяется алгоритм мэппинга, с использованием которого и находится лучший мэппинг. Для демонстрации разрабатываемого подхода в статье предлагается жадный алгоритм мэппинга, ориентированный на архитектуру с топологией 3-х-мерного тора. Подход находится в стадии разработки. Экспериментальная реализация этапов предлагаемого подхода демонстрируется на примере тестовых приложений для суперкомпьютера IBM Blue Gene/P.

Статья состоит из введения, 6 разделов и заключения. В разделе 1 дается обзор подходов к построению мэппинга. Общее описание предлагаемого подхода дано в разделе 2. В разделе 3 приводится краткое описание метода определения характерных коммуникационных матриц. В разделе 4 описывается модель сверточной нейронной сети для классификации коммуникационных матриц и выбора на этой основе алгоритма мэппинга. В разделе 5 подробно рассматривается предлагаемый метод построения мэппинга. В разделе 6 приводятся результаты проведенных вычислительных экспериментов с использованием разработанных алгоритмов. В заключении подводятся итоги исследования и предлагаются направления дальнейших работ.

1. Обзор алгоритмов мэппинга параллельных приложений

Алгоритмы, направленные на анализ коммуникационного поведения программ и решение задачи мэппинга, активно исследуются. Регулярно появляются новые публикации, посвященные различным аспектам решения этой актуальной задачи.

Алгоритмы, решающие задачу мэппинга, можно, условно, разделить на 3 типа: алгоритмы, работающие с графами, эволюционные алгоритмы и улучшающие алгоритмы. К первой группе можно отнести алгоритмы для нахождения изоморфизма графов разными способами, например, рекурсивное разбиение графа, поиск в ширину, «лучший — первый», RCM. При рекурсивном разбиении графов [4, 5] граф параллельного приложения и граф вычислительный системы рекурсивно разрезаются на два подграфа по какому-либо свойству графа. На выходе из рекурсии происходит отображение вершин графа приложения на граф вычислительной системы [2, 3]. При поиске в ширину во время первого прохода граф помечается. Далее на каждом шаге алгоритма на граф вычислительный системы отображается вершина графа приложения с наибольшим количеством еще не отображенных соседей [3]. Одним из простейших алгоритмов вида «лучший — первый» является жадный алгоритм [2, 5]. Случайным образом выбирается процесс, который помещается в коммуникационную сеть. Потом из списка выбирается сосед, с которым есть взаимодействие и ставится рядом в сети. На вход RCM алгоритму подается коммуникационная матрица, которая рассматривается как матрица смежности графа приложения. RCM алгоритм переставляет столбцы и строки матрицы таким образом, чтобы уменьшить ширину ленты матрицы [2, 3]. Эволюционными алгоритмами являются, например, различные генетические алгоритмы, разыскивающие мэппинг с помощью «направленного» перебора, отсекая заведомо плохие решения. Улучшающие алгоритмы предполагают наличие какого-либо начального решения и направлены на улучшение полученного решения. Например, итерационный алгоритм может менять расположение случайных процессов для получения лучшего локального решения [2].

2. Схема предлагаемого подхода

Основная идея предлагаемого подхода заключается в следующем.

Рассмотрим параллельную MPI-программу, запущенную на n процессах. Под трассой T параллельной программы будем понимать совокупность трасс процессов: $T = \{T_1, T_2, \ldots, T_n\}$. Трасса процесса — это последовательность событий: $T_i = \{s_1, s_2, \ldots, s_k\}$. Событие — это вызов MPI-функции. Учитывая особенности организации коммуникационных систем передачи данных, реализованных в BlueGene/P, будем рассматривать только функции передач типа точка-точка. Каждое событие характеризуется временем наступления, объемом сообщения, номером процесса-получателя. Под коммуникационной матрицей в данной работе понимается целочисленная квадратная матрица, размер которой равен числу процессов в MPI-программе. Элемент в позиции (i, j) равен суммарному размеру передаваемых сообщений (в байтах) от процесса с номером i процессу с номером j. Матрица может рассматриваться на временном промежутке $[t_1, t_2]$. В этом случае элемент в позиции (i, j) равен суммарному объему сообщений, переданных от процесса i к процессу j, передача которых была инициирована в момент времени $t \in [t_1, t_2]$.

Формально постановку задачи можно определить следующим образом. Пусть параллельная программа задана множеством параллельных процессов $p_i : P = \{p_1, p_2, \ldots, p_n\}$. Известна трасса программы T. Архитектура вычислительной системы представлена мно-

жеством узлов Q. Требуется найти отображение $F : P \to Q$ множества $P = \{p_1, p_2, \ldots, p_n\}$ процессов параллельной программы на множество $Q = \{q_1, q_2, \ldots, q_n\}$ узлов вычислительной системы. Такое отображение называется мэппингом. Большинство существующих алгоритмов мэппинга решают более частную задачу. В этой постановке рассматривается граф параллельного приложения $G = (V(G), E(G))$ и граф вычислительной системы $H = (V(H), E(H))$. Вершинами графа G являются процессы. Ребра соответствуют наличию коммуникаций между ними и имеют вес, соответствующий интенсивности этих коммуникаций. Вершины графа H соответствуют узлам вычислительной системы, ребра соответствуют наличию каналов для передачи сообщений.

Требуется найти отображение $T : V(G) \to V(H)$ вершин графа приложения на вершины графа вычислительной системы на котором достигается минимум функции:

$$\sum_{i,j=1}^{N} d(T(p_i), T(p_j)) \cdot w(p_i, p_j),$$

где $V(G) = \{p_1, p_2, \ldots, p_n\}$, $V(H) = \{q_1, q_2, \ldots, q_n\}$, $w(p_i, p_j)$ — вес ребра между вершинами p_i и p_j в графе G, $d(a, b)$ — кратчайшее расстояние между вершинами a и b графа H.

Таким образом, для использования алгоритмов мэппинга при решении общей задачи требуется по трассе приложения предъявить граф приложения. Для представления графа параллельного приложения используется коммуникационная матрица, которая является матрицей смежности графа.

Для описания коммуникационного поведения параллельной программы будем использовать коммуникационные матрицы, соответствующие некоторым заданным интервалам времени. Конкретный способ построения таких матриц может выбираться, исходя из имеющихся инструментов анализа поведения параллельных программ. В работе рассматривается метод сбора коммуникационных матриц для MPI-программ на основе их трасс. Сбор трасс реализован неинвазивным методом с помощью переопределения MPI-функций и их подключением к исполняемому коду путем динамической линковки. Учитываются вызовы `MPI_Send`, `MPI_Isend`, `MPI_Sendrecv`, как наиболее часто используемые вызовы для передач типа точка-точка. Коллективные операции, выполнение которых осуществляется в BlueGene/P выделенной коммуникационной сетью, в нашем случае рассматривать не будем. Данный метод не требует наличия исходного кода программы.

Альтернативным способом построения мэппинга может быть подача на вход алгоритму мэппинга коммуникационной матрицы, соответствующей какому-то интервалу времени, вместо суммарной коммуникационной матрицы программы, так как эта матрица может лучше представлять коммуникации в приложении, существенно влияющие на его производительность.

Как было показано выше, различные алгоритмы мэппинга могут по-разному влиять на время выполнения параллельного приложения по сравнению с используемым принятым по умолчанию стандартным алгоритмом мэппинга. Предположим, что у нас есть априорные знания об алгоритмах мэппинга и типах приложений, которые они ускоряют. Пусть эти знания представлены в виде набора классов коммуникационных матриц и подходящих алгоритмов мэппинга для каждого класса. Тогда данную коммуникационную матрицу мы можем отнести к одному из этих классов и на основе этого выбрать подходящий алгоритм мэппинга. Для классификации заданной коммуникационной матрицы предлагается использовать сверточную нейронную сеть, заранее обученную на различных выборках мат-

риц. Основным преимуществом нейронной сети является быстрота подбора подходящего алгоритма мэппинга.

Таким образом, предлагается следующая идея (рис. 1). Временная шкала выполнения параллельной программы разбивается на отрезки, называемые фазами. Метод выделения фаз описывается в разделе 3. Для каждой фазы вычисляется коммуникационная матрица, представляющая шаблон взаимодействия процессов в рамках этой фазы. Среди полученных коммуникационных матриц фаз на основе заданного критерия определяется одна матрица. Проводится классификация этой матрицы с помощью сверточной нейронной сети. Модель сверточной нейронной сети описана в разделе 4. В зависимости от класса, к которому была отнесена матрица, выбирается соответствующий данному классу алгоритм мэппинга. Пример одного из таких алгоритмов описан в разделе 5.

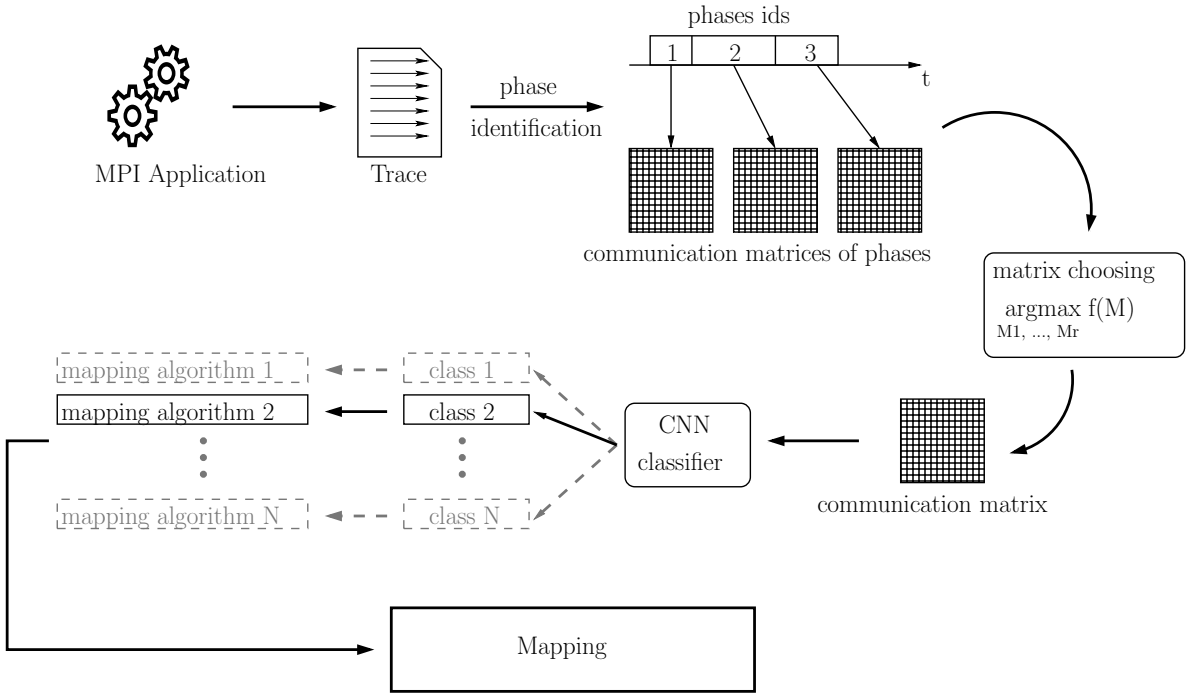


Рис. 1. Схема предлагаемого подхода к решению задачи мэппинга

3. Метод выбора коммуникационного шаблона для построения мэппинга

Опишем более детально этапы выделения фаз и выбора подходящей фазы в качестве коммуникационного шаблона для построения мэппинга. Описание метода приведем в соответствии с работой, выполненной авторами ранее [6]. Идея метода заключается в использовании вейвлет-преобразований для выделения характеристик коммуникационных матриц. Коммуникационные матрицы при этом будем представлять, как изображения. Применим к временному ряду построенных таким образом изображений метод поиска изображений, описанный в работе [7]. Метод использует двумерное вейвлет-преобразование Хаара. В указанной работе найдены оптимальные веса, на которые должны умножаться признаки после вейвлет-преобразования для оптимизации поиска изображения. Предложенные веса используются в предлагаемом нами методе. Метод состоит из следующих шагов: сбор трассы параллельного приложения; разбиение трассы равными временными промежутками длины Δt и построение коммуникационной матрицы для каждого промежутка (построение после-

довательности коммуникационных матриц); приведение матриц к стандартному размеру $m \times m$; применение к матрицам двумерного вейвлет-преобразования Хаара; умножение матриц на специальные поправочные веса; кластеризация полученных матриц; определение фаз в построенной последовательности классов.

Трасса параллельной программы разбивается на равные промежутки фиксированной длины Δt (Δt является параметром метода). Для каждого промежутка строится коммуникационная матрица. Имеем последовательность коммуникационных матриц.

Матрицы уменьшаются до размера $m \times m$ следующим образом: вся матрица разбивается равномерно на $m \times m$ клеток, в каждую клетку заносится усредненное значение всех элементов, попавших в данную клетку. Это делается по двум причинам: во-первых, используемое на следующем шаге вейвлет-преобразование требует, чтобы размер входной матрицы m являлся степенью двойки; во-вторых, при больших размерах матрицы вейвлет-преобразование и последующая кластеризация будут занимать много времени.

Далее, к полученным матрицам применяется двумерное вейвлет-преобразование Хаара. Оно заключается в применении одномерного вейвлет-преобразования Хаара сначала к каждой строке матрицы, а затем к каждому столбцу результирующей матрицы.

Одномерное преобразование Хаара заключается в выполнении операций усреднения и разности $log_2 N$ раз над массивом длины N. Сначала вычисляются средние по каждой паре чисел и записываются в первую половину выходного массива, затем вычисляются разности между первым числом в паре и средним и записываются во вторую часть массива (эти разности еще называют коэффициентами детализации). Затем та же самая процедура применяется к первой половине массива. И так далее до длины массива, равной единице.

После вейвлет-преобразования элементы матрицы умножаются на поправочные веса. Это умножение проводится с целью учета более важных коэффициентов после вейвлет-преобразования при подсчете евклидового расстояния между матрицами (во время кластеризации). Выбор этих весов $w[i], i = 0, 1, \ldots, 5$ подробно описан в статье [7].

Элемент матрицы с координатами (i, j) умножается на вес $w[bin(i, j)]$, где:

$$bin(i, j) = min(max(level(i), level(j), 5),$$

$$level(i) = [log_2(i + 1)].$$

Для отнесения похожих матриц к одному классу проводится их кластеризация. Матрицы рассматриваются как векторы размерности m^2. В качестве расстояния используется евклидово расстояние. Для кластеризации рассматривались методы: k-средних, DBSCAN.

После кластеризации имеем последовательность классов, к которым были отнесены матрицы временных промежутков анализируемой программы. В этой последовательности выделяются максимально длинные отрезки постоянства класса. Назовем эти отрезки фазами. Таким образом, каждая фаза состоит из стоящих рядом отрезков длины Δt, отнесенных к одному классу. Следовательно, в рамках фазы коммуникационное поведение не меняется или меняется незначительно. Если в программе коммуникационное поведение будет существенно меняться, каждый «шаблон» взаимодействия окажется в отдельной фазе.

Для каждой из полученных фаз строится коммуникационная матрица этой фазы. Среди них выбирается одна матрица в качестве коммуникационного шаблона приложения для построения мэппинга. Критерий выбора в общем виде имеет вид: $\underset{M \in \{M_1, M_2, \ldots, M_r\}}{argmax} f(M)$, где r — число фаз; $M_1, M_2, \ldots, M_r$ — матрицы фаз, $f : \mathbb{R}^{n \times n} \to \mathbb{R}$ — некоторая числовая

функция от матрицы. Т. е. требуется выбрать матрицу на которой достигается максимум функции f. В данной работе в качестве функции f рассматривается сумма всех элементов матрицы. Такой выбор обоснован тем, что фаза с наибольшим объемом переданных данных (сумма элементов ее матрицы будет больше сумм элементов матриц остальных фаз) предположительно представляет те коммуникации в программе, которые наиболее сильно влияют на производительность. Выбор функции f подлежит дальнейшему исследованию.

4. Сверточная нейронная сеть для классификации коммуникационных матриц

Классификацию коммуникационных матриц выполним на основе нейросетевого подхода. За основу нейросетевой модели была выбрана нейронная сеть VGG-16 [8], которая хорошо зарекомендовала себя в работе с изображениями, а коммуникационную матрицу можно представить, как изображение с одним каналом.

Построенная нейросетевая модель состоит из трех блоков свертка-свертка-макспулинг (conv-conv-maxpool), в каждом из которых свертка происходит с ядром размера 3×3, а пулинг с ядром размера 4×4 в первых двух блоках и 2×2 в последнем блоке. Количество фильтров в сверточных слоях равно 16, 32, 64 для соответствующих блоков. После трех блоков, описанных выше, следует персептрон с одним внутренним слоем.

Схема предложенной нейронной сети представлена на рис. 2.

Для построения и обучения модели использовался фреймворк Keras [9], который разработан специально для ускорения разработки и простоты использования нейронных сетей. Keras работает поверх одной из мощных библиотек Tensorflow, CNTK, Theano. В данной работе была использована библиотека Tensorflow.

Входными данными для обучения являются матрица и ее класс (целое число). Перед началом обучения к матрицам применяется ранговое преобразование по строкам: по каждой строке создается массив пар (номер элемента в строке, элемент), после чего полученный массив сортируется по элементу в порядке возрастания. Далее каждому элементу в исходной строке присваивается 0, если элемент был 0, и номер элемента в массиве пар, если элемент не 0. Нумерация начинается с единицы. Таким образом, исходная строка преобразуется с помощью рангового преобразования и имеет следующие свойства: все элементы будут в пределах от 0 до размерности матрицы, и если элемент i в исходной строке был больше элемента j, то тоже самое соотношение будет в полученной строке.

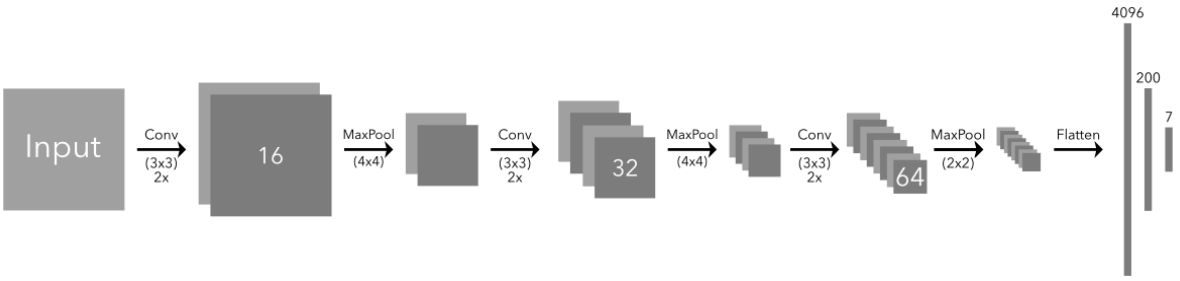


Рис. 2. Архитектура предложенной нейросетевой модели

Такая нейросетевая модель решает задачу классификации параллельных приложений по коммуникационной матрице, вследствие чего можно применить наилучший алгоритм мэппинга для полученного класса, а не подбирать алгоритм вручную.

5. Алгоритм мэппинга для HPC-систем с топологией 3D тор

Общая идея разработанного алгоритма мэппинга состоит в размещении наиболее часто взаимодействующих процессов на как можно более близких узлах параллельной вычислительной системы. Коммуникации между процессами определяются с помощью коммуникационной матрицы параллельного приложения, которая подается на вход алгоритму. Для каждого процесса алгоритм пытается найти 6 наиболее коммуницирующих соседей, если такие есть. Количество соседей выбрано не случайно: алгоритм создавался для коммуникационной сети типа 3-х-мерный тор, а в данной сети каждый узел имеет соединение с 6 узлами. После нахождения подходящих соседей для каждого процесса, алгоритм определяет новые координаты для процессов в трехмерном торе и строит мэппинг для данного параллельного приложения.

На рис. 3 представлена схема предложенного алгоритма.

Опишем предложенный алгоритм подробнее. На вход алгоритму подается коммуникационная матрица параллельной программы. Каждая строка матрицы сортируется, и создается внутренняя структура, которая называется мэппинг-вектор: матрица, размером $N \times 6$, где каждой строке под номером i выбирается 6 соседей, с которыми у данного процесса наибольший объем коммуникаций.

Далее генерируется внутреннее представление трехмерного тора, и процессы сортируются по множествам Process, Last, Done:

- множество Process содержит процессы, у которых есть хотя бы одна коммуникация;
- множество Last содержит процессы, которые не участвуют в коммуникациях точка-точка;
- множество Done содержит процессы, которые уже расставлены на трехмерный тор.

После распределения процессов по множествам, используя мэппинг-вектор и построенные множества, алгоритм расставляет процессы на трехмерный тор: выбирается самый нагруженный по объему сообщений процесс и ставится в 3D решетку вместе со своими соседями, с которыми у данного процесса наибольший объем коммуникаций. Все эти процессы заносятся в множество Done, так как они уже расставлены на коммуникационной решетке, а «инициирующий» процесс удаляется из множества Process. Далее выбирается один из соседей и те же шаги повторяются для него, если он еще содержится в множестве Process.

В конце расстановки по свободным местам в решетке распределяются процессы из множества Last, так как процессы без коммуникаций можно поставить в любое место решетки. Когда в множествах Process и Last не остается элементов, расстановка завершается. Для вывода мэппинга трехмерный тор переводится в одномерный массив пар (номер процесса, координаты в решетке) и сортируется по номеру процесса. После этого вторые значения в паре (координаты) выводятся в файл, который и будет мэппингом параллельной программы.

Сложность алгоритма порядка $O(N^2)$, где N — количество процессов.

6. Вычислительный эксперимент

6.1. Пример определения фаз и выбора коммуникационного шаблона

Для демонстрации применимости метода определения коммуникационного шаблона рассмотрим искусственное приложение, состоящее из последовательного вызова двух тестов из набора NAS Parallel Benchmarks: LU и CG [10]. Эксперимент проводился на си-

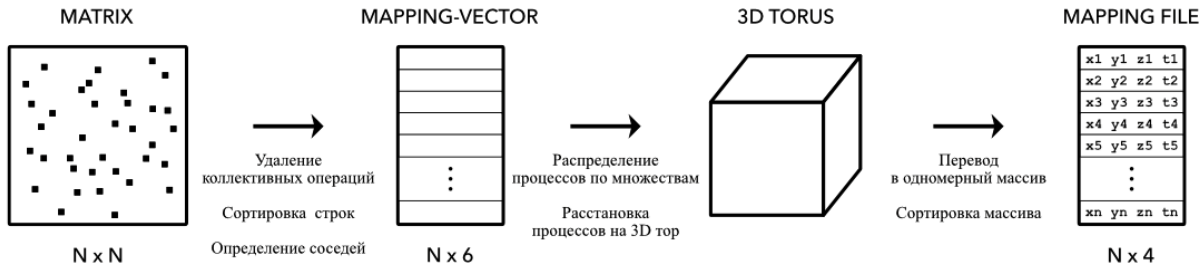
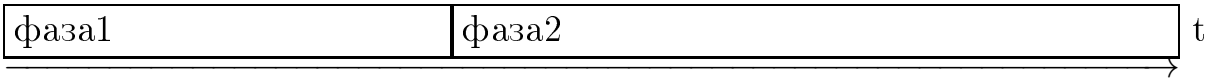
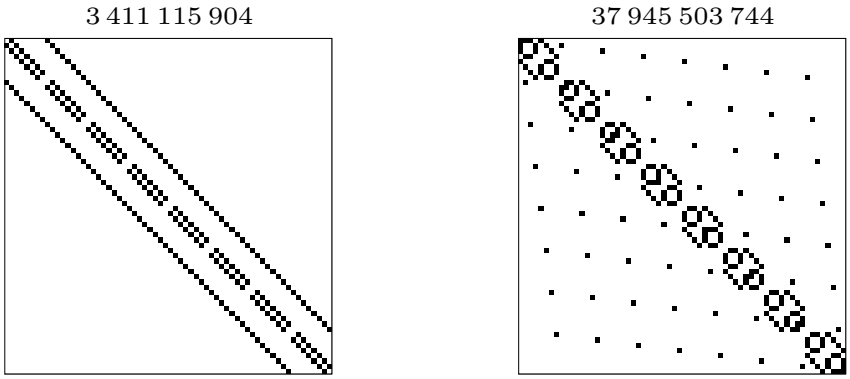


Рис. 3. Схема предложенного алгоритма мэппинга

стеме IBM Blue Gene/P на 64 узлах. Полученные фазы и их коммуникационные матрицы представлены на рис. 4. Числа над матрицами указывают сумму всех элементов матрицы. Видно, что вторая матрица имеет сумму значительно больше первой. Следовательно, она должна выбираться в качестве коммуникационного шаблона для построения мэппинга.



а) Фазы на временной шкале выполнения программ



б) Фаза 1 в) Фаза 2

Рис. 4. Пример определения фаз

6.2. Пример применения нейросетевой классификации тестовых коммуникационных матриц

Для тестирования нейросетевой модели и алгоритма мэппинга разработан генератор матриц 7 классов, модели которых представлены на рис. 5. Такие матрицы были выбраны после анализа нескольких статей (например, [11]) и немного упрощены для более легкого построения большого количества матриц. Среди представленных классов есть как похожие друг на друга матрицы, так и совершенно различные, что требуется для тестирования нейросетевой модели.

При генерации матриц случайно выбирались матрицы, которые случайным образом «зашумлялись»: добавлялись случайные ненулевые числа в случайные ячейки матрицы. Процент шума для каждой матрицы выбирался также случайным образом.

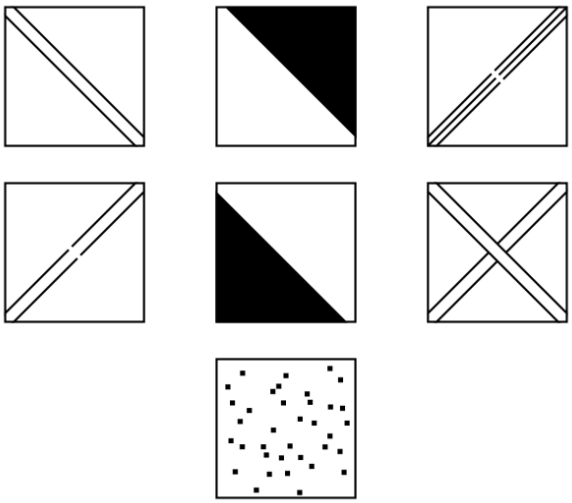


Рис. 5. Классы матриц, которые были использованы для обучения и тестирования

Для исследования эффективности разработанного алгоритма мэппинга использовались несколько классов матриц из числа описанных выше. С этой целью были разработаны тестовые программы, которые выполняют пересылки таким образом, что коммуникационная матрица таких программ имеет вид случайной матрицы, матрицы с двумя обратными диагоналями и зашумленной матрицы с двумя обратными диагоналями. На рис. 6 представлено сравнение времени выполнения параллельных приложений при различных алгоритмах мэппинга. Для каждой из трех тестовых программ использовалось два мэппинга: стандартный (XYZT) и полученный с помощью алгоритма greedy. Время работы параллельной программы со стандартным мэппингом принято за 100%. Из рисунка видно, что для данных классов матриц удалось получить ускорение на 36%, 26% и 56% соответственно для каждого из классов, относительно стандартного мэппинга.

Рис. 6. Сравнение времени выполнения параллельных приложений при различных алгоритмах мэппинга

Нейросетевая модель тестировались на матрицах размера 256×256, а эксперимент для алгоритма мэппинга проводился для программ, запущенных на 256 узлах соответственно.

Заключение

В статье описан разрабатываемый подход к повышению эффективности параллельных приложений. Разработанные в рамках подхода методы позволяют выделить наиболее характерные для заданного приложения паттерны коммуникационного поведения, провести классификацию выделенных паттернов, выбрать лучший алгоритм мэппинга из заданного набора и построить мэппинг. Подход к решению задачи мэппинга на основе использования нейросетевых моделей свертки не имеет аналогов. В настоящее время реализованы основные этапы предложенного подхода: реализован метод построения характерного паттерна для заданного приложения, реализован нейросетевой подход для классификации коммуникационных матриц, предложен новый алгоритм мэппинга для HPC-систем с архитектурой 3-х-мерного тора. Проведенные вычислительные эксперименты с использованием реализованных методов демонстрируют возможность их применения не только для решения задачи мэппинга в рассматриваемой постановке, но позволяют решать вопрос об их применимости и для других задач, связанных с исследованием поведения параллельных приложений на высокопроизводительных системах.

Горизонт применения и дальнейшего развития предложенного подхода широкий. Прежде всего, планируется расширение множества рассматриваемых классов коммуникационных матриц и алгоритмов мэппинга, используемых для предложенной нейросетевой модели. Будет проведено исследование конкретных параллельных приложений с использованием предложенного подхода. Будут уточняться уже реализованные методы.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-07-01053.

Литература

1. Список TOP500. URL: `https://www.top500.org/` (дата обращения: 10.04.2019).
2. Hoefler T., Snir M. Generic topology mapping strategies for large-scale parallel architectures // Proceedings of the international conference on Supercomputing (ICS '11) (Tucson, Arizona, USA, May, 31–June, 04, 2011). ACM, 2011. P. 75–84. DOI: 10.1145/1995896.1995909.
3. Sreepathi S., D'Azevedo E., Philip B., Worley P. Communication Characterization and Optimization of Applications Using Topology-Aware Task Mapping on Large Supercomputers // Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering (ICPE '16) (Delft, The Netherlands, March, 12–16, 2016). ACM, 2016. P. 225–236. DOI: 10.1145/2851553.2851575.
4. Wu J., Xiong X., Lan Z. Hierarchical task mapping for parallel applications on supercomputers // The Journal of Supercomputing. 2015. Vol. 71, no. 5. P. 1776–1802. DOI: 10.1007/s11227-014-1324-5.
5. Hoefler T., Jeannot E., Mercier G. An overview of topology mapping algorithms and techniques in high-performance computing // High-Performance Computing on Complex Environment. 2014. P. 75–94. DOI: 10.1002/9781118711897.ch5.
6. Шубин М.В., Попова Н.Н. Анализ поведения параллельных MPI-программ на основе фаз межпроцессного взаимодействия // Суперкомпьютерные дни в России: Труды международной конференции (Москва, 24–25 сентября 2018 г.). Москва: Издательство МГУ, 2018. С. 662–672.

7. Jacobs C., Finkelstein A., Salesin D. Fast multiresolution image querying // Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH '95) (Los Angeles, CA, August, 1995). ACM, 1995. P. 277–286. DOI: 10.1145/218380.218454.

8. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556. URL: https://arxiv.org/abs/1409.1556 (дата обращения: 10.04.2019).

9. Документация фреймворка Keras. URL: https://keras.io/ (дата обращения: 10.04.2019).

10. Описание приложений NPB. URL: https://www.nas.nasa.gov/publications/npb.html (дата обращения: 10.04.2019).

11. Asanovic K., Bodik R., Catanzaro B., Gebis J., Husbands P., Keutzer K., Patterson D., Plishker W., Shalf J., Williams S., Yelick K. A View of the Parallel Computing Landscape // Communications of the ACM. 2009. Vol. 52, no. 10. P. 56–67, DOI: 10.1145/1562764.1562783.

Попова Нина Николаевна, к.ф.-м.н., доцент, кафедра суперкомпьютеров и квантовой информатики, Московский государственный университет имени М.В. Ломоносова (Москва, Российская Федерация)

Козлов Михаил Владимирович, студент, кафедра суперкомпьютеров и квантовой информатики, Московский государственный университет имени М.В. Ломоносова (Москва, Российская Федерация)

Шубин Михаил Витальевич, студент, кафедра суперкомпьютеров и квантовой информатики, Московский государственный университет имени М.В. Ломоносова (Москва, Российская Федерация)

DOI: 10.14529/cmse200103

NEURAL NETWORK APPROACH FOR MAPPING OF PARALLEL APPLICATIONS

© 2020 N.N. Popova, M.V. Kozlov, M.V. Shubin

Lomonosov Moscow State Universisty

(GSP-1, Leninskie Gory 1, Moscow, 119991 Russia)

E-mail: popova@cs.msu.su, rat.taurus@gmail.com, mihshub@gmail.com

Received: 12.11.2019

The article is about the problem of improving parallel applications efficiency. It proposes an approach to solve this problem. The approach aims to reduce communication overheads of data exchange between parallel program processes during its execution on high-performance computing system. Growing number of computer nodes leads increasing impact of the communication overhead on application performance. As a sequence the problem of parallel processes allocation to hardware nodes (mapping problem) becomes very actual. The new approach for mapping problem is proposed in this work. The key characteristic of the approach is to extract a communication pattern by phase analysis of application using a convolutional neural network to fast choosing the most relevant mapping algorithm for extracted pattern.

Investigation of results of point-to-point communications between parallel program processes is used to build a communication pattern. The application timeline is broken into equal intervals. Communication patterns are created for each interval. Then Haar 2D wavelet transform is applied to these patterns for generating features. Features are clustered, after that, timeline is splitted into phases. Each phase has its own communication pattern.

The selection of the most relevant mapping algorithm is performed by using convolutional neural network. It supposes some knowledge about different types of parallel applications and most suitable mapping algorithms. This knowledge must be represented by a set of pattern classes (classes of matrices), each class has the mapping algorithm, which fits best for application with this type of pattern. This set can be a training sample for the neural network. Thus the neural network classifies an input application communication pattern and finds a mapping algorithm for the application.

The stages of proposed approach are implemented in this work. This implementation is demonstrated for some tests.

Keywords: high performance computing systems, topology-aware mapping, communication pattern, convolution neural network.

FOR CITATION

Popova N.N., Kozlov M.V., Shubin M.V. Neural Network Approach for Mapping of Parallel Applications. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2020. Vol. 9, no. 1. P. 36–49. (in Russian) DOI: 10.14529/cmse200103.

References

1. TOP500 List. Available at: `https://www.top500.org/` (accessed: 10.04.2019).
2. Hoefler T., Snir M. Generic topology mapping strategies for large-scale parallel architectures. Proceedings of the international conference on Supercomputing (ICS '11) (Tucson, Arizona, USA, May, 31–June, 04, 2011). ACM, 2011. P. 75–84. DOI: 10.1145/1995896.1995909.
3. Sreepathi S., D'Azevedo E., Philip B., Worley P. Communication Characterization and Optimization of Applications Using Topology-Aware Task Mapping on Large Supercomputers. Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering (ICPE '16) (Delft, The Netherlands, March, 12–16, 2016). ACM, 2016. P. 225–236. DOI: 10.1145/2851553.2851575.
4. Wu J., Xiong X., Lan Z. Hierarchical task mapping for parallel applications on supercomputers. The Journal of Supercomputing. 2015. Vol. 71, no. 5. P. 1776–1802. DOI: 10.1007/s11227-014-1324-5.
5. Hoefler T., Jeannot E., Mercier G. An overview of topology mapping algorithms and techniques in high-performance computing. High-Performance Computing on Complex Environment. 2014. P. 75–94. DOI: 10.1002/9781118711897.ch5.
6. Shubin M.V., Popova N.N. Study of the behavior of parallel MPI-programs based on phases of interprocess communication. Russian Supercomputing Days: Proceedings of the international conference (Moscow, Russia, September, 24–25, 2018). Moscow, Moscow State University, 2018. P. 662–672. (in Russian)
7. Jacobs C., Finkelstein A., Salesin D. Fast multiresolution image querying. Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH '95) (Los Angeles, CA, August, 1995). ACM, 1995. P. 277–286. DOI: 10.1145/218380.218454.
8. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556. Available at: `https://arxiv.org/abs/1409.1556` (accessed: 10.04.2019).

9. Keras Documentation. Available at: `https://keras.io/` (accessed: 10.04.2019).

10. NAS Parallel Benchmarks. Available at: `https://www.nas.nasa.gov/publications/npb.html` (accessed: 10.04.2019).

11. Asanovic K., Bodik R., Catanzaro B., Gebis J., Husbands P., Keutzer K., Patterson D., Plishker W., Shalf J., Williams S., Yelick K. A View of the Parallel Computing Landscape. Communications of the ACM. 2009. Vol. 52, no. 10. P. 56–67. DOI: 10.1145/1562764.1562783.

УДК 004.75 **DOI: 10.14529/cmse200104**

ЦИФРОВОЙ ПРОЕКТ И ПЛАТФОРМА ДЛЯ РАБОТЫ С НИМ*

© 2020 Е.В. Биряльцев[1,2], М.Р. Галимов[2], Д.Е. Демидов[3,4], А.М. Елизаров[3,4]

[1]*Институт прикладных исследований Академии наук Республики Татарстан (420111 Республика Татарстан, Казань, ул. Лево-Булачная, д. 36А),*

[2]*ООО «Градиент технолоджи» (420111 Республика Татарстан, Казань, ул. Большая Красная, д. 63),*

[3]*Казанское отделение Межведомственного суперкомпьютерного центра РАН — филиал ФГУ ФНЦ «НИИ системных исследований» РАН (420111 Республика Татарстан, Казань, ул. Лобачевского, д. 2/31),*

[4]*Казанский (Приволжский) федеральный университет (420111 Республика Татарстан, Казань, ул. Кремлевская, д. 35)*

E-mail: Igenbir@yandex.ru, glmvmrt@gmail.com, dennis.demidov@gmail.com, amelizarov@gmail.com

Поступила в редакцию: 09.08.2019

Обоснована актуальность представления сложных наукоемких цифровых проектов в виде направленного графа, объединяющего в единый цифровой объект входные, выходные и промежуточные данные с программными модулями преобразования информации. Предложен метод манипулирования с таким представлением в виде облачной интернет-платформы. Последняя включает в себя центральный сервер приложений и хранилище (репозиторий), обеспечивающие хранение алгоритмов и данных, регистрацию и сопровождение пользователей, коммуникации между ними, а также учет использования ими алгоритмов и данных при решении прикладных задач. Работа с алгоритмами и данными происходит в исполняемой среде, загружаемой при присоединении к платформе либо на машину пользователя, либо на виртуальную машину в облачном кластере. Эта среда обеспечивает создание, модификацию и использование алгоритмов (в том числе, из множества предлагаемых стандартных), которые решают конкретные прикладные задачи конкретного пользователя. Взаимодействие пользователя с сервером и репозиторием осуществляется через веб-интерфейс или толстый клиент на локальной или виртуальной машинах. Представлен работающий прототип названной платформы, функционирующий с использованием суперкомпьютерных технологий и системы виртуализации рабочих столов. Прототип включает в себя инструменты создания программных средств на основе графово-модульной архитектуры и коммуникационные сервисы для участников. Он позволяет выполнять вычислительные графы в высокопроизводительной среде, обеспечивать регистрацию интеллектуальной собственности в галерее и осуществлять биллинг ее использования. Приведены реализованные примеры возможного использования платформы в геофизических исследованиях и в области государственного управления.

Ключевые слова: интернет-платформа, программная платформа, облачные сервисы, прикладное программное обеспечение, разработка программного обеспечения, автоматизация бизнес-процессов.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Биряльцев Е.В., Галимов М.Р., Демидов Д.Е., Елизаров А.М. Цифровой проект и платформа для работы с ним // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 1. С. 50–68. DOI: 10.14529/cmse200104.

*Статья рекомендована к публикации Программным комитетом Международной конференции «Суперкомпьютерные дни в России — 2019».

Введение

В современном обществе проектная деятельность приобретает все большее значение. Этому способствуют: тенденция к гиперсегментации рынков, требующая разработки уникальных предложений для уникальной группы пользователей; возрастающая роль инновационной экономики, создающей принципиально новые товары и услуги; освоение междисциплинарных ниш, требующих комбинации технологий, ранее совместно не используемых; бизнес-подход к управлению данными (Data-driven approach), включающий в производственный цикл стадию изучения и восстановления закономерностей в исследуемых данных, а также другие тенденции, характеризующие отличия информационного общества от постиндустриального.

Основная доля себестоимости уникальных продуктов и услуг приходится на проектную часть, в отличие от массового продукта, где стоимость проектной деятельности распределяется на большую серию однотипных объектов, а себестоимость определяется эффективностью производства. Поэтому дальнейшее снижение себестоимости в обсуждаемой деятельности в настоящее время требует снижения себестоимости именно проектной части.

Проектная деятельность, в отличие от производственной, является преимущественно цифровой и потенциально хорошо подлежит автоматизации. Вместе с тем проектная деятельность имеет существенные отличия от массового производства, и ее автоматизация должна производиться с учетом этих отличий.

Считая любой проект частным, наукоемким случаем инновации вообще, мы можем воспользоваться методологией, предложенной Чесбро в [1] и последующих работах, и констатировать, что построение эффективной проектной деятельности, замкнутой в рамках одной иерархической структуры, невозможно: для эффективной реализации проекта необходимы привлечение различных участников и организация между ними эффективного рыночного взаимодействия.

Проектная деятельность, с точки зрения экономической науки, является многосторонним рынком с ассиметричной информацией, что порождает высокие транзакционные издержки, барьеры и риски. Автоматизация деятельности такого рода должна быть направлена в первую очередь на уменьшение транзакционных издержек. Многосторонние рынки и рынки с асимметричной информацией являются в последнее время приоритетным предметом изучения экономической науки. Для эффективной работы на таких рынках предложены концепция многосторонней платформы и концепция обеспечения доверия на основе распределенного реестра. Важным фактором для успешного превращения объекта в рыночный товар является его коммодитизация, т.е. представление в объективном измеримом виде.

Настоящая статья организована следующим образом. В первом разделе рассмотрены подходы, которые в настоящий момент используются для автоматизации проектной деятельности. Второй раздел посвящен описанию подхода, предложенного авторами, а также созданного прототипа программной платформы. В третьем разделе представлен ряд примеров использования созданного прототипа для решения реальных практических задач. В заключении дана краткая сводка результатов, полученных в работе, и указаны направления дальнейших исследований.

1. Используемые подходы

Отметим, что автоматизация проектной деятельности активно развивается в настоящее время в следующих направлениях:

1. *Графовые вычисления.* Представление вычислительного процесса в виде направленного ациклического графа и предоставление пользователям возможности самостоятельно, без помощи программиста, конструировать и видоизменять этот граф в настоящее время широко применяются в такой области, как Data Mining. Здесь можно отметить, например, такие системы анализа данных, как RapidMiner Studio и KNIME Analytics Platform, а также интеграционные платформы Mule ESB и Apache NiFi. Графовое представление обработки геофизических данных также давно применяется в нефтегазовом секторе в продуктах таких компаний, как Schlumberger, Paradigm, Roxar и ряда других. Например, компания SAP представила недавно продукт Data Hub, позволяющий строить графы извлечения информации из различных источников и построения аналитических обрабатывающих графов в рамках одной оболочки. Перечисленные программные продукты можно рассматривать как средства быстрой разработки и изменения инструментов извлечения знаний и интеграции информационных систем.

2. *Средства коллективной разработки программного обеспечения.* Исторически первыми такими программами были платформы, обеспечивающие версионное хранение исходных кодов проектов (Github, Gitlab и Bitbucket). В настоящий момент они представляют собой комплексные сервисы по разработке и управлению программными проектами и, кроме хранения кода, обеспечивают командную работу, документирование проектов, интеграцию со средствами непрерывной сборки и тестирования приложений. Сегодня компания Amazon представляет аналогичный сервис AWS Step Functions для организации координации компонентов распределенных приложений и микросервисов в виде графических схем рабочих потоков. Крайне интересно выглядят два новых проекта Pallium.network и SingularityNET, которые ставят своей целью построение распределенной в интернете коммерческой сети для решения задач, связанных с машинным обучением. Предполагается, что сторонние пользователи смогут подключать свои программные решения в качестве отдельных узлов такой сети и затем предлагать их для коммерческого использования остальным участникам.

3. *Автоматизация учета интеллектуальной собственности и биллинга ее использования.* В настоящее время технологии распределенного реестра, первоначально использованные в криптовалютах, применяются также для учетных операций с токенизированными активами вообще и операциями с интеллектуальной собственностью в частности. Например, в упомянутых выше Pallium.network и SingularityNET для фиксации авторства используется блокчейн. Платформа Ethereum дает возможность строить на ее основе умные контракты, исполняющиеся по наступлению определенных событий, что позволяет осуществлять взаимные микроплатежи за использование ресурсов в автоматическом режиме.

Предлагаемый ниже подход является интеграцией перечисленных выше подходов.

2. Предлагаемый подход

2.1. Представление цифрового проекта

Для автоматизации проектной деятельности необходимо формализовать понятие и представление проекта. Мы будем рассматривать проект как последовательность действий, приведших к некоторому результату, с целевой функцией, определяющей соответствие результата цели проекта. Хорошо известны нотации IDEF0, DFD и аналогичные, применяемые для описания последовательности действий вида, приведенного на рис. 1 (слева). В этой диаграмме под действием понимается далее любое не специфицируемое действие с материальными или цифровыми объектами, осуществляемое управляющим механизмом, под которым может пониматься автомат или человек.

Для цифрового проекта, все действия которого выполняются с цифровыми объектами, мы можем использовать аналогичный формализм (рис. 1, справа), конкретизируя понятия действия как абстрактного вычислительного алгоритма, формализованного в виде λ-функции, ограничений как глобальных переменных, входящих в заданную целевую функцию, а также управляющего механизма как совокупности тела функции и ее параметров, задаваемых в процессе выполнения.

Аналогично методологии SADT сложные функции иерархически декомпозируются на более простые, вплоть до элементарных действий, а выходные данные предыдущих узлов могут являться для последующих узлов входными данными, ограничениями или управляющими параметрами.

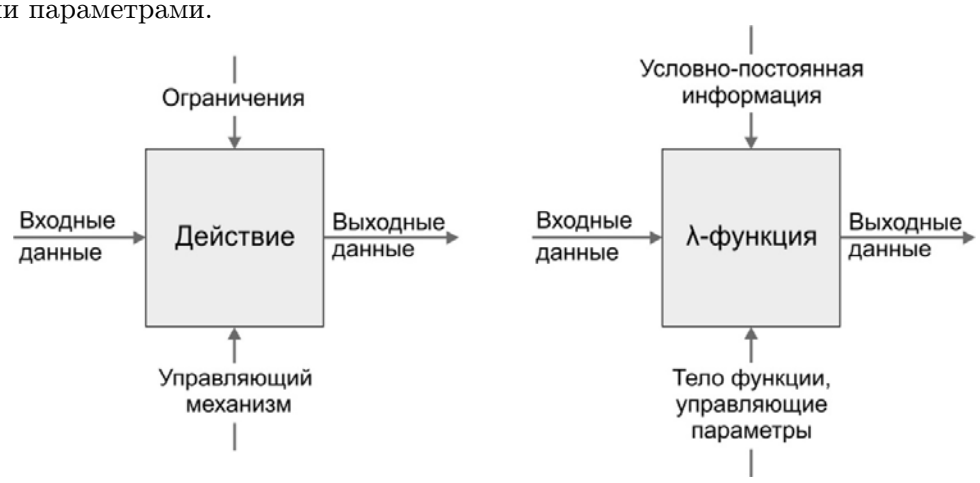


Рис. 1. IDEF0-нотация и представление проекта как λ-функции

Выполненный проект представляет собой конкретный набор выходных параметров, полученный в результате применения алгоритма, выбранного пользователем, в условиях имеющихся ограничений и параметров, им заданных. Ключевым отличием предлагаемого подхода от нотации IDEF0 является физическое представление проекта в виде графа, узлами которого является исполняемый программный модуль со специфицированными управляющими параметрами, а дугами служат наборы данных. Объект «Проект» физически сохраняется в таком виде вместе с данными и исполняемыми модулями и может быть повторно, полностью или частично, исполнен оболочкой, поддерживающей такое представление.

Предлагаемое представление проекта изменяет понятие «Прикладное программное обеспечение» (ППО). Традиционно ППО было организовано в виде предметно-ориентированных автоматизированных рабочих мест (АРМ), которые агрегировали известный функционал. Выполнение проекта предусматривало перемещение информации между АРМ. В предлагаемом нами подходе АРМ отсутствуют, функции могут выбраны из библиотек или предыдущих проектов или написаны пользователем и включены в граф проекта по мере необходимости. В результате любой проект является исполняемым в относительно простой оболочке, не требует наличия у пользователя никаких дополнительных программных средств, что существенно упрощает его коллективное выполнение. Для повторяющихся проектов или их составных частей могут быть выделены типовые проекты, снабженные интерфейсом пользователя и соответствующей документацией.

2.2. Особенности платформы для операций с цифровыми проектами

В настоящее время признанным механизмом эффективной реализации многосторонних сделок являются платформы [2]. Предложенное выше унифицированное представление проекта частично коммодитизирует проект как предмет сделки, однако сама сделка по выполнению проекта отличается от обычных актов купли-продажи. Понимая проект как наукоемкую инновацию, можно применить саарбрюккенскую модель [3] инновации как многостадийного процесса (рис. 2), включающего научную, инженерную, предпринимательскую стадии и бизнес-стадию.

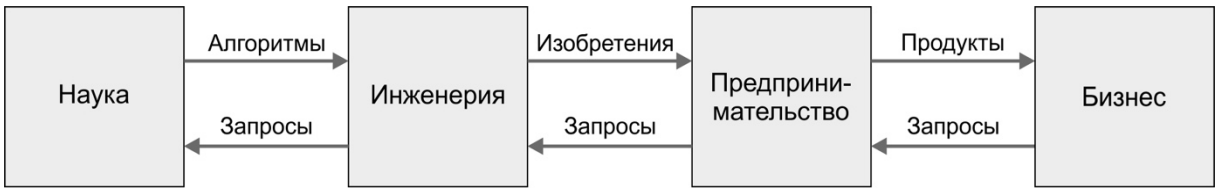


Рис. 2. Саарбрюккенская модель инновационного процесса

Из этой диаграммы видно, что процесс выполнения проекта состоит как минимум из 3-х типов сделок (если не считать сделки внутри стадий между однотипными исполнителями), а характер этих сделок различен. Таким образом, чтобы существенно снизить транзакционные издержки, платформа должна поддерживать все три типа внутри проектных сделок, а также финальную сделку между бизнесом и конечным потребителем проекта. Для цифрового проекта мы можем специфицировать объект сделки по стадиям следующим образом.

1. *На научной стадии* создаются цифровые модели объектов и процессов, выраженные в виде систем уравнений, а также методы решения этих систем для различных случаев. Все эти объекты выражаются в виде алгоритмических модулей, имеющих входные, выходные и управляющие параметры, а также условно-постоянную информацию. От инженерной стадии поступает запрос на модификацию модели или метода решения, в обратном направлении поступают варианты модулей. Возможна также ситуация, когда модуль создается в инициативном порядке и далее предлагается инженерной стадии.

2. *На инженерной стадии* создаются цепочки модулей как метод решения некоторой группы инженерных задач. Цепочка собирается из готовых или заказных модулей и может быть привязана к графическому интерфейсу пользователя (GUI) для демонстрации

возможностей. Инициатива в создании изобретения может принадлежать инженерной стадии либо формироваться в виде запроса от предпринимательской стадии.

3. *На предпринимательской стадии* изобретение адаптируется к конкретному потребителю (группе потребителей) и представляет собой технологию, доступную для применения линейным специалистам. На этой стадии специфицируются источники исходной информации, при необходимости реализуются адаптеры-преобразователи этой информации из исходного вида в тот вид, который требует применяемый метод, а также определяются характеристики метода для данного потребителя и т. д. Цепочка модулей привязывается к промышленному GUI для повышения эффективности использования.

4. *Бизнес-стадия.* Для индустриального общества на этой стадии производится масштабирование производства и доставки товара/услуги потребителю. Для постиндустриальной стадии производство может ограничиться единственным внедрением, и созданный продукт будет уникальным. Вместе с тем созданный продукт может использоваться в более крупных проектах в виде подпроекта или стать прототипом для похожих проектов. В первом случае бизнес-стадия представлена компанией-интегратором, использующей частные цепочки для создания более крупной компании, во втором случае распространение может идти через партнеров в различных предметных и территориальных секторах рынка. В обоих этих случаях для повторного использования продукта может потребоваться его модификация на стадиях: предпринимательской (адаптация к новой структуре информации, терминологии, квалификации и структуре персонала заказчика), изобретательской (изменение графа для учета качественно новой исходной информации, формирования дополнительных выходных данных) или научной (для адаптации моделей или алгоритмов).

GUI к цепочкам может быть реализован по портальной технологии. Для каждого узла разрабатывается собственное диалоговое окно, включающее визуализацию всех данных и задание параметров обработки. Интерфейс к цепочке в целом собирается из отдельных окон с заданием правил перехода между ними.

Помимо многосторонности проектная деятельность обладает асимметрией информации на всех стадиях, представители соседних стадий плохо понимают специфические проблемы друг друга, а через стадию – не понимают вообще. В связи с этим возникает недоверие к любым результатам, представляемым одной стадии другой. Мы можем разделить эту проблему на две оставляющие: недоверие на основе опасения фальсификации и недоверие на основе сомнения в компетентности.

В последнее время проблема невозможности подделки получила решение в виде методов распределенного реестра [4]. Нетрудно заметить, что графовое представление проектов легко адаптируется к применению таких методов. Мы можем проиндексировать каждый узел его хеш-функцией, включающей хеши кода узлов, параметров, входных и выходных данных. Объемные данные при этом могут выстраиваться в иерархическую структуру хешей, известную как дерево Меркла. Так как входные данные для любого узла получаются из выходных данных предыдущих узлов, узлы получаются связанными по данным, и вся цепочка однозначно зафиксирована цепочкой хешей. Для окончательной фиксации цепочки последний хеш можно опубликовать в публичном реестре или связать со следующим случайно выбранным проектом, исключая, таким образом, возможность искажения результатов, полученных впоследствии.

Для проверки компетенции исполнителей возможно проведение внешней или внутренней экспертиз, заключающихся в частичном или полном пересчете графа. Во-первых, таким образом можно пересчитать некоторые или все результаты и убедиться, что они получены декларируемой цепочкой. Во-вторых, возможен пересчет результатов с другими параметрами обработки для получения уверенности в субоптимальности полученного результата. Таким же образом можно провести сравнение структурно различных вариантов проекта, например, для демонстрации преимущества новой структуры графа или реализации отдельных модулей. Выполненный таким образом проект может фиксировать не только финальную цепочку расчетов, но и проверенные, а также забракованные результаты, для чего необходимо зафиксировать концы всех забракованных вариантов в отдельном блоке. Промежуточные данные забракованных веток можно не хранить, а ограничиться только их хешами (рис. 3).

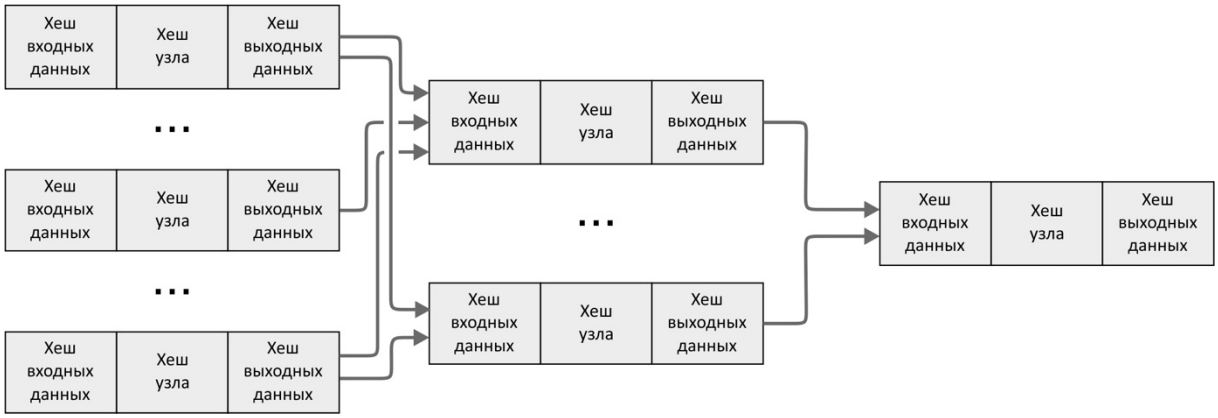


Рис. 3. Представление проекта цепочкой хеш-ссылок

2.3. Функциональный состав платформы

Платформа как комплекс программных продуктов должна выполнять определенный обязательный набор функций (ядро) и поддерживать применение сервисов, упрощающих взаимодействие пользователей.

Узел А на данном уровне абстракции представляет собой одновременно структуру данных, хранящихся на платформе, и исполняемый алгоритм в виде скрипта на интерпретируемом языке или контейнере. Узел должен быть выполнен в REST-архитектуре и быть исполняемым на любом кластере платформы.

К ядру относятся следующие функции.

1. Управление выполнением цепочек – осуществляется аналогом планировщика задач, который получает готовые к выполнению модули, связанные с входными и условно-постоянными данными и параметрами, и распределяет их по исполняющим узлам. Планировщик может использовать различные политики назначения исполнителя, вплоть до биржевых механизмов (отдает тому, кто меньше запросит денег). Планировщик также ведет журналирование хода выполнения работ и учет затраченных ресурсов.

2. Управление хранением данных – выполняет подсистема хранения, осуществляющая хранение узлов, графов и их данных в едином формате. Система поддерживает связность графов на основе частного блокчейна, обеспечивает ведение версий выполняемых и выполненных проектов, резервирование и страйпинг данных, служебные функции.

3. Коммуникации между пользователями. Подсистема поддерживает ведение форумов, чатов, архивов, маркетплейсов и других средств коммуникации пользователей, включая аудио-видео конференции.

4. Администрирование пользователей и ресурсов – управление созданием, модификацией, удалением пользователей и ресурсов, назначением им прав и лимитов.

5. Взаимодействие системы с пользователем (общий GUI). Все предыдущие функции доступны через API и едины на платформе. Для взаимодействия с пользователями платформы предоставляется базовый интерфейс пользователя, возможно в нескольких вариантах.

Сервис представляет собой систему взаимодействующих публичных проектов, доступ к которым регламентируется правилами присоединения к сервису. Состав сервисов может наращиваться пользователями платформы, однако следующие сервисы представляются наиболее важными:

1. Поисково-рекомендательный сервис – производит поиск модулей, графов, пользователей, ресурсов, а также их агрегацию и ранжирование по релевантности запросу с учетом запросного контекста. Впоследствии может быть развит в диалоговую систему типа личного голосового помощника.

2. Биллинговый сервис – на основе журналирования использования модулей и графов формирует сводные отчеты по использованным ресурсам (возможно, с тарификацией) для потребителей и владельцев ресурсов.

3. Мастера построения и отладки:

3.1. Графов и модулей. Включает в себя интерактивный конструктор и отладчик графов, редактор-отладчик текстов модулей на скриптовых языках (по типу MATLAB) к процедурным узлам, конструктор SQL-запросов для логических узлов. Поддерживает ведение версий на стадии разработки-отладки.

3.2. GUI. Визуальный конструктор и отладчик интерфейсных модулей к графам. Интерфейс к модулю является экранным плагином, предоставляющим пользователю возможность в графическом или текстовом виде просматривать входные, выходные и условно-постоянные данные, а также задавать параметры исполняемого модуля. GUI может охватывать более одного узла, в этом случае пользователю предоставляются доступ ко всем данным узлов для просмотра и возможность задания параметров всех узлов. Отдельные GUI могут группироваться в сценарии работы с графом. Сценариев также может быть больше одного.

4. Управления проектами. Осуществляются планирование и отслеживание хода проектов на платформе менеджерами. Данные о ходе работ берутся непосредственно с платформы и не требуют ручного ввода пользователями. Можно строить прогноз окончания выполнения работ и моделировать его при изменении ресурсов проекта.

В дальнейшем очевидными направлениями развития сервисов являются:

- оптимизационный сервис: граф, для которого определена функция качества результата, можно рассматривать как функцию, требующую оптимизации; возможно построение сервиса, который будет заниматься перебором вариантов управляющих параметров с использованием алгоритмов оптимизации;
- обучающий сервис: возможно проигрывание графа в специальном режиме, направленном на усвоение пользователем особенностей работы с этим графом;

- сервис отчетов и публикаций, оформляющий результаты работы в человекочитаемый вид;
- сервис извлечения знаний: массив однотипных выполненных проектов (или подпроектов), особенно вместе с забракованными вариантами, является обучающей выборкой для извлечения знаний о том, как надо решать подобные задачи; извлеченные знания могут использоваться для оптимизационного сервиса, контроля действий пользователя или самостоятельного выполнения подпроектов искусственным интеллектом;
- сервис взаиморасчетов: биллинговая система ограничивается сбором информации о затраченных и представленных ресурсах и формировании комплексных счетов, расчеты при этом производятся через внешние системы; учитывая надежную фиксацию всех выполненных операций, мы можем вести расчеты во внутренней криптовалюте, в том числе, с использованием смарт-контрактов.

2.4. Прототип платформы

Согласно описанной выше концепции был разработан прототип платформы, содержащий следующие основные компоненты:

- регистрирующий узел (centralregnode);
- управляющий кластер (controlnode);
- вычислительный кластер (computecluster);
- клиентские приложения (cloudview).

Регистрационный узел — это точка регистрации различных управляющих и вычислительных кластеров для обеспечения интеграционных процедур и создания единой точки доступа. Кроме того, регистрационный узел обеспечивает учет пользователей для всех кластеров, подключенных к данной платформе, а также размещение для всех проектов общедоступных объектов.

Управляющий кластер — это программно-аппаратная инфраструктура, реализующая основные сервисы платформы и предоставляющая к ним удаленный доступ в интернет-среде. Исполнение ресурсоемких программ производится на подключаемых вычислительных кластерах.

Вычислительный кластер — это высокопроизводительная программно-аппаратная инфраструктура, предназначенная для выполнения ресурсоемких пользовательских программ (скриптов). Вычислительные кластеры могут быть как интегрированными в управляющие кластеры, так и являться кластерами общего пользования, входящими в общедоступные вычислительные сети.

Клиентские приложения — это набор программного обеспечения, который обеспечивает взаимодействие пользователя с сервисами управляющего кластера платформы. Клиентские приложения могут подключаться к любым управляющим кластерам при наличии прав у соответствующих пользователей. Для поиска доступных управляющих кластеров может использоваться регистрационный узел.

Архитектура платформы позволяет проводить гибкую настройку в зависимости от потребностей пользователей и создавать на ее основе как закрытые площадки внутри компаний, так и открытые – в интернете. Кроме того, имеется потенциальная техническая возможность интеграции различных разрозненных площадок в одну, что часто требуется при проведении сложных комплексных работ с множеством исполнителей.

Рассмотрим подробнее архитектуру управляющего кластера — основного элемента платформы. Он состоит из:

- центрального сервера приложения;
- реляционного хранилища;
- объектного хранилища;
- сервисов.

Центральный сервер приложений является единой точкой доступа к сервисам, предоставляемым данным управляющим кластером, а также обеспечивает:

- API&CLI для работы сервисов;
- среду размещения/регистрации сервисов платформы;
- организационное взаимодействие пользователей с платформой.

Реляционное хранилище предназначено для хранения служебной информации платформы, структуры и метаинформации графов обработки, хранения информации сервисов, а также непосредственного хранения данных небольших объемов.

Объектное хранилище предназначено для хранения пакетов данных больших размеров. Непосредственное хранение осуществляется в специализированном key-value архиве, в качестве которого выступает база данных PostgreSQL 10. Данные хранятся в виде байтовых массивов в записях таблиц. Для снижения нагрузки на единичные таблицы данные распределяются по группе таблиц согласно первым символам хеша (ключа) хранения. Предусмотрена возможность разбиения данных на блоки определенного размера и хранения в виде цепочки таких блоков.

Сервисы — это программные модули, развернутые на центральном сервере приложений и реализующие различные функциональные возможности управляющего кластера платформы. Сервисы взаимодействуют с основными компонентами управляющих и вычислительных кластеров платформы программным способом и предоставляют пользователю возможность оперировать функционалом платформы.

Таким образом, управляющий кластер позволяет хранить и обрабатывать данные различных типов и размеров, а также обеспечивать работу группы пользователей.

Важной особенностью платформы является возможность проведения вычислительно нагруженных и ресурсоемких работ по моделированию и обработке данных различных исследований. Для реализации этого принципа управляющий кластер прототипа был интегрирован с высокопроизводительной GPU-системой и системой удаленного VDI-доступа, программно-аппаратная архитектура которых описана в [5]. Для проведения распределенных вычислений поддерживается система управления общими ресурсами и выполнения групп задач SLURM, а для выполнения локальных задачи используется система управления NQ.

В рамках работ по формированию прототипа был реализован ряд клиентских приложений, которые обеспечивают:

- администрирование платформы;
- функционирование личного кабинета пользователя;
- управление проектом (редактор и менеджер проекта);
- коллективную работу на основе проведения дискуссий и обмена графами обработки и данными на форумах и галереях платформы.

Все клиентские приложения разработаны в виде desktop-клиента с использованием языка программирования Python 3 и технологии PyQt 5. В связи с этим для обеспечения

возможности работы с данными больших размеров основная деятельность пользователей происходит удаленно на площадке VDI-кластера. Начаты исследования по разработке веб-клиента для организации взаимодействия с сервисами платформы. Для возможности расширения функциональности клиентских приложений сторонними пользователями предусмотрена система плагинов. Плагин позволяет изменять как визуальное представление приложения пользователю, так и расширять его производственные возможности.

Разработанный прототип использован для реализации прикладных пилот-проектов в различных предметных областях, некоторые из которых описаны ниже.

3. Пилотные проекты

В настоящее время параллельно с разработкой прототипа платформы реализуется несколько прикладных пилот-проектов с ее использованием, что позволяет отладить функционал и юзабилити платформы во взаимодействии с реальными конечными пользователями. Ниже дано краткое описание трех проектов из различных предметных областей, использующих технологии высокопроизводительных вычислений.

3.1. Локализация микросейсмической эмиссии

Нефтегазовый сектор экономики переживает сегодня существенную трансформацию как по объектам разработки, так и методам обработки геофизической и геологической информации. Поскольку значительная часть традиционных запасов углеводородов исчерпана, начинается активное освоение так называемых трудноизвлекаемых запасов, в число которых входят сланцевые углеводороды. Последние отличаются, в частности, отсутствием привязки залежей к куполообразным погребенным структурам-ловушкам, в связи с чем традиционные методы сейсморазведки не работоспособны для разведки сланцевых залежей. Поэтому сейчас активно разрабатываются новые методы, основанные, в частности, на локации повышенной микросейсмической эмиссии, характерной для трещиноватых зон с повышенным скоплением углеводородов.

Технологии локализации зон повышенной микросейсмической эмиссии являются быстроразвивающейся областью прикладной науки и технологии, базирующихся на высокопроизводительных вычислениях для численного моделирования распространения сейсмических волн в сложных средах и применения статистических методов большой размерности для обработки полевых сигналов.

В качестве пилот-проекта на прототипе платформы был реализован алгоритм локализации микросейсмической эмиссии [6], в котором реализованы такие вычислительно нагруженные задачи, как решение динамической задачи для вязкоупругой среды методом конечных элементов с размерностью порядка сотен миллионов ячеек и методы дисперсионного анализа с размерностью ковариационных матриц порядка десятков тысяч (см. также [7, 8]). Отличительной особенностью метода является использование априорной информации в качестве регуляризатора, что требует перестройки алгоритма под различные виды априорной информации, не позволяет создать универсальный АРМ для решения данной задачи и требует корректировки программного обеспечения. Для решения этой задачи был разработан базовый граф, включающий основные алгоритмы и используемый в качестве типового решения. Платформенная реализация работы с графовыми представлениями проектов позволяет геофизику, реализующему конкретный проект, произвести реконфигурацию графа, добавляя или исключая отдельные модули, а также

написать собственный модуль, учитывающий особенности проекта, и включить его в проект, не прибегая к услугам профессиональных программистов.

Взаимодействие геофизика с графом производится через пользовательский интерфейс, набранный из типовых экранных элементов, позволяющих осуществлять ввод управляющих параметров и просмотр входных и выходных данных каждого узла в виде графиков, диаграмм, карт и трехмерных динамических объектов. Граф проекта и интерфейс одного из модулей представлены на рис. 4.

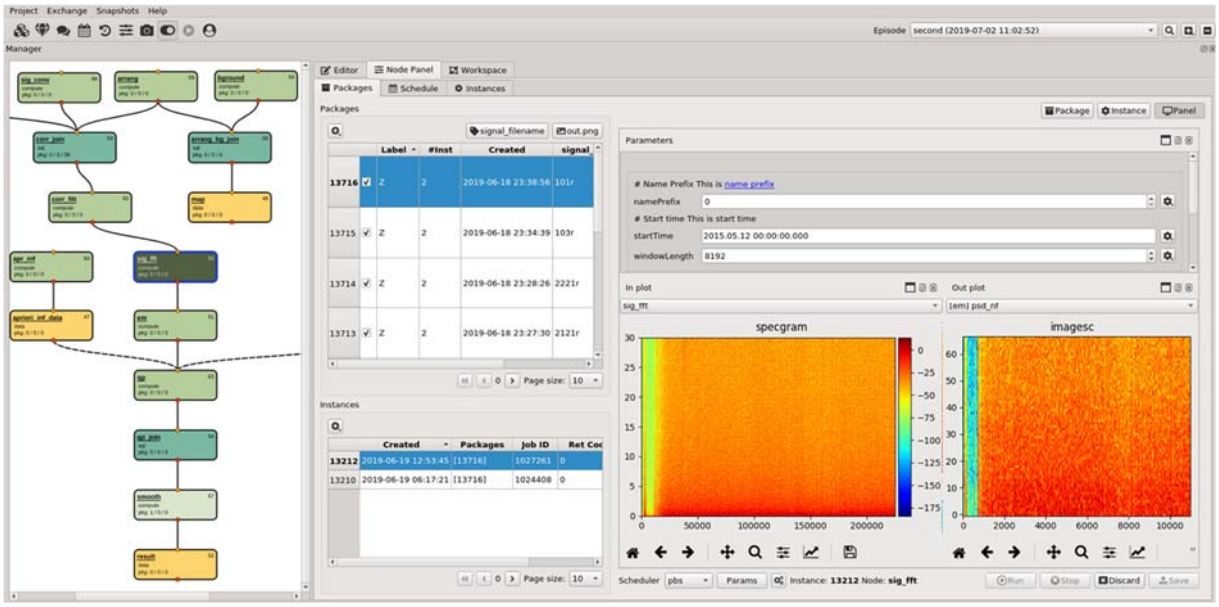


Рис. 4. Вычислительный граф и интерфейс пользователя проекта

3.2. Доступ к источникам информации

Развиваемый нами подход предложен в качестве основы для решения нестандартных аналитических задач Ситуационному центру (СЦ) Республики Татарстан [9]. Такие аналитические задачи отличаются необходимостью анализа заранее неизвестного набора информации в сжатые сроки программирующими пользователями–аналитиками СЦ. В качестве пилотного проекта был реализован доступ к данным сайта «Открытый Татарстан» [10], содержащего разнообразные отчеты ведомств. Этот сайт имеет API для доступа к данным, однако его формат ориентирован на выдачу отдельных файлов-отчетов по периодам в формате JSON, анализируемый период при этом выносился API в название файла в неунифицированном формате. Доступ к отчету осуществляется по его идентификатору, не несущему семантической информации о содержимом. Таким образом, для анализа временного ряда какого-либо параметра необходимо установить идентификаторы отчетов, извлечь по идентификатору, а затем провести разбор и склейку нескольких файлов с использованием библиотек парсинга сплошного текста и формата JSON, что затруднительно для программирующих пользователей.

Для доступа к данным сайта «Открытый Татарстан» разработан скрипт на языке Python, параметризованный по идентификатору отчета, извлекающий набор отчетов и преобразующий его в реляционную таблицу. Для набора отчетов установлено соответствие идентификатора и названия отчета, скрипт доступа с соответствующим идентификатором был представлен как источник данных и помещен в галерею доступа к данным.

Организован нечеткий поиск с элементами семантического анализа по названию источника данных и названиям выходных переменных, что позволяет пользователю-аналитику искать наборы данных по их смыслу. Данные возможности представлены на рис. 5.

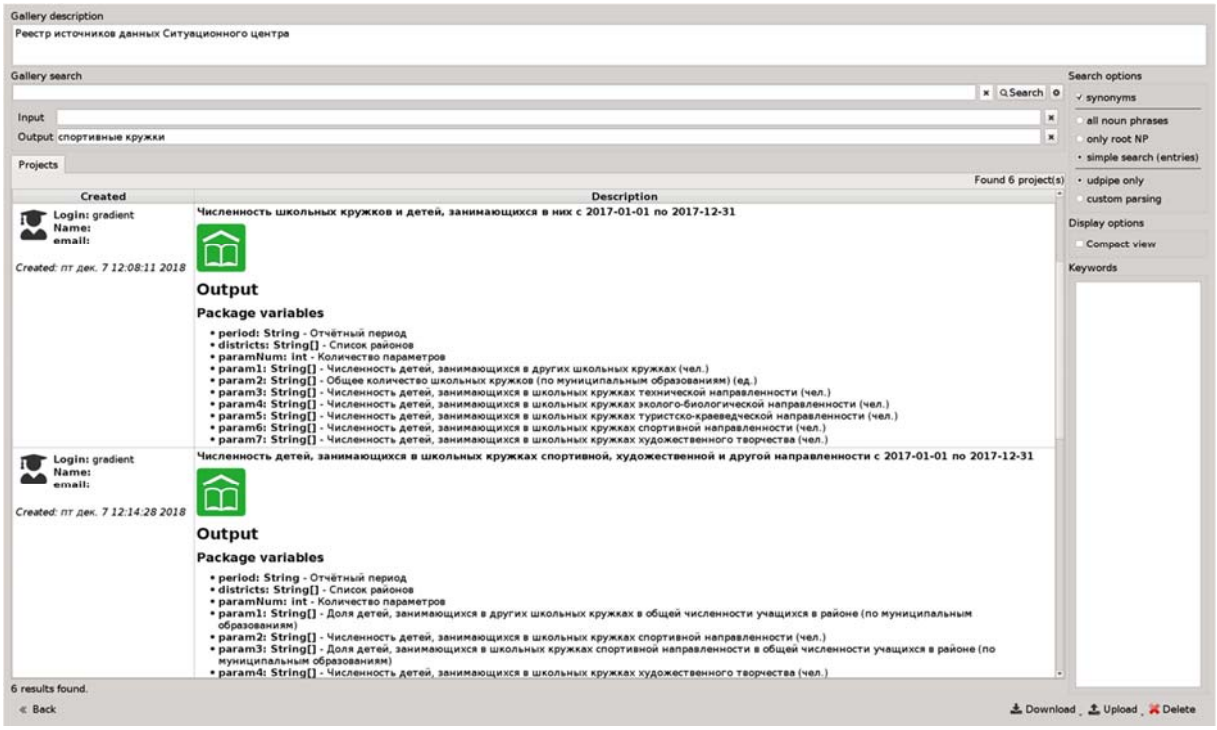


Рис. 5. Каталог скриптов доступа к данным и интерфейс нечеткого поиска

Найденные источники данных могут быть перенесены аналитиком в рабочую область платформы. При необходимости совместного анализа источники данных соединяются в смысле оператора JOIN по общим ключам в единую таблицу. Анализ данных статистическими методами из библиотеки статистических модулей, например, методом главных компонент, позволяет определить характер распределения объектов, выделить основные кластеры и отклоняющиеся объекты и провести другие аналитические исследования выбранных данных (рис. 6).

Механизм доступа к источникам данных, реализованный на платформе, позволяет скрыть от аналитика реализацию хранения данных в конкретных источниках, разгрузив его от преобразования данных. Вместе с тем графовое представление проекта позволяет аналитику самостоятельно набрать источники данных для анализа, а также выбрать аналитический метод или их комбинацию, не прибегая к помощи прикладного программиста.

3.3. Сервис семантического поиска

Как отмечалось выше, одним из важных сервисов платформы является семантический поиск, в том числе, экспертов в узких предметных областях. В ситуации отсутствия большого количества пользователей на прототипе платформы был реализован эмулятор такого сервиса, позволяющий искать экспертов, используя аннотации их публикаций в научно-технической печати и различные заявки на проекты. Реализованная система базируется на внешних источниках данных о научно-технической активности ученых и специалистов и состоит из двух процессов, описанных ниже. При анализе физико-математических документов применялись технологии, развитые, например, в [11–13].

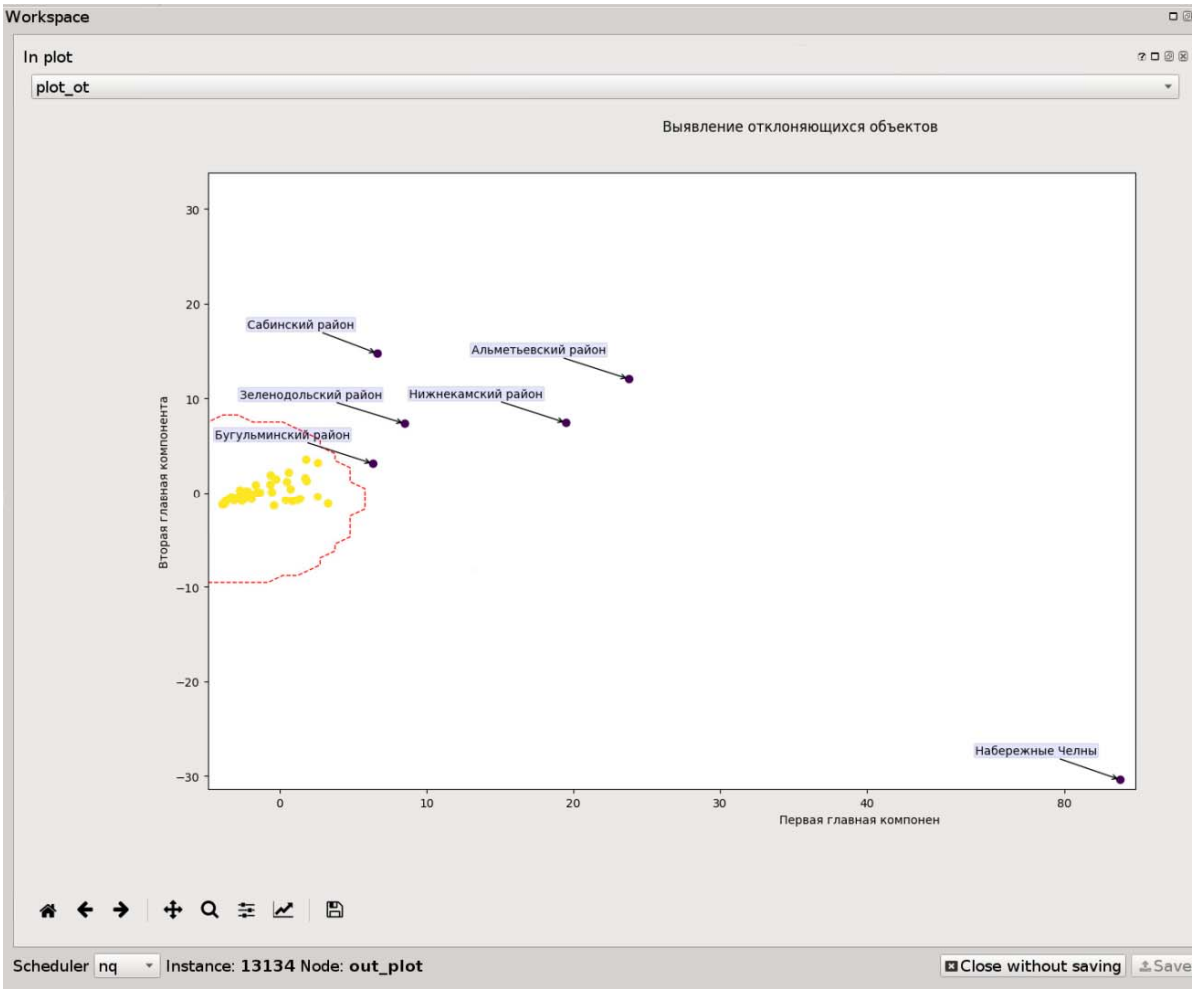


Рис. 6. Пример интерфейса аналитика

Первый процесс извлекает сведения о проведенных или планируемых научно-технических разработках из различных источников данных. В модельном примере в качестве источников были использованы заявки на конкурс 50 лучших идей Татарстана (около 1000 заявок) и данные о публикациях сотрудников Казанского федерального университета (около 56 тыс. публикаций). Информация извлекалась из источников, унифицировалась по формату, фильтровалась по информативности (отбрасывались неинформативные аннотации и некорректные заявки) и преобразовывалась в единую базу знаний о научно-технической активности. Такой процесс выполняется периодически администратором системы по мере изменения данных в старых источниках таких данных или добавления новых источников.

Второй процесс связан с обработкой пользовательских запросов. Пользователь вводит описание интересующей его предметной области в строку запроса (рис. 7), после чего система разворачивает его в набор синонимов и по всем получившимся вариантам выполняет поиск в базе знаний по аннотациям выполненных работ. Каждой найденной записи ставится в соответствие ее вес, при этом учитываются давность публикации и близость найденной формулировки к первоначально введенной пользователем. Найденные аннотации группируются по авторам, веса аннотаций суммируются и автору присваивается суммарный индекс, отражающий степень его экспертного опыта, соответствующего запросу, введенному пользователем. Для группы экспертов, найденных описанным способом, может быть выполнен просмотр аннотаций их научно-технических проектов и статей, а

также сформирован адрес электронной почты для установления контактов и дальнейшего уточнения информации.

Описанная реализация системы поиска экспертов может быть развита в двух направлениях – как пользовательская система с подключением более насыщенных источников информации, в частности, данных таких общероссийских научных информационных систем, как eLibrary, баз данных научных фондов (РФФИ, РНФ), Роспатента и других организаций, а также как сервис внутри самой платформы, позволяющий искать авторов прикладных проектов, выполненных непосредственно на ней.

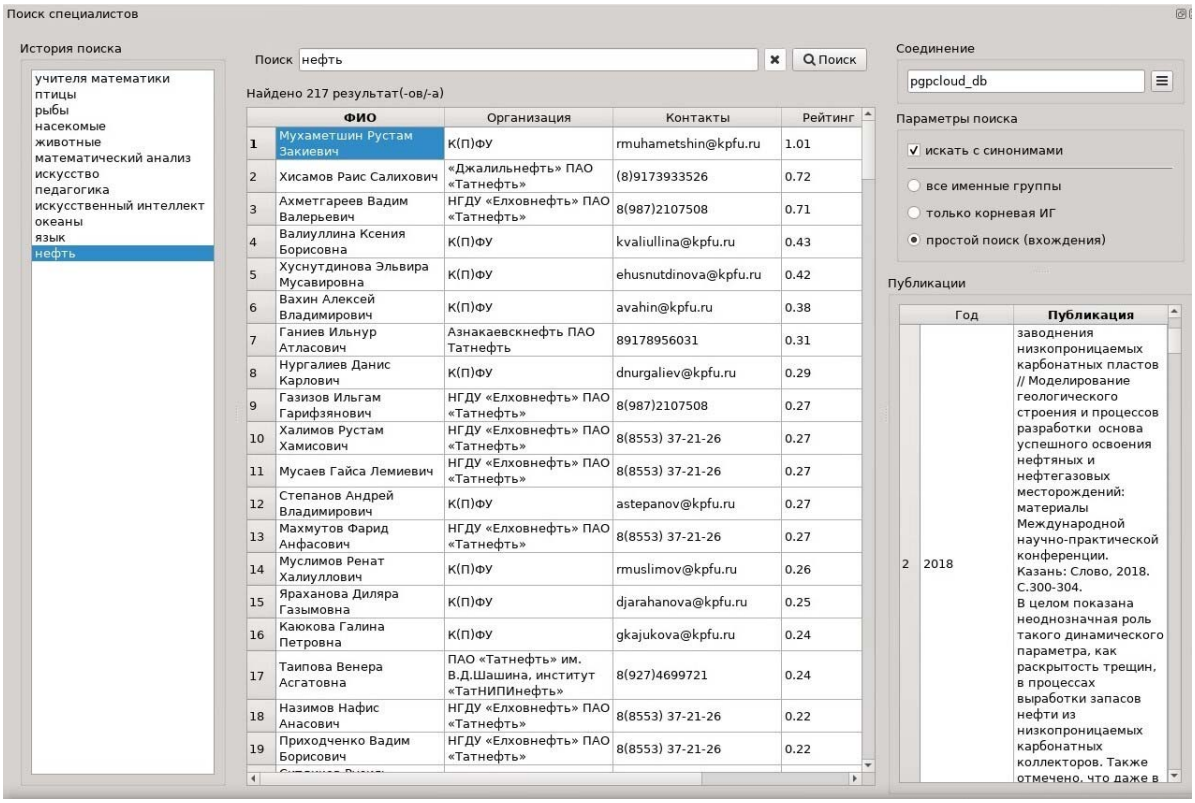


Рис. 7. Интерфейс поиска

Заключение

Проблемы и задачи, обсуждаемые выше в статье, могут быть решены путем использования комплексной программной интернет-платформы, включающей в себя инструменты создания, изменения и выполнения графа цифрового проекта, средства контроля хода работ и биллинга использования материальных и нематериальных ресурсов, а также коммуникационные средства для взаимодействия участников.

Предложенная форма представления цифрового проекта и реализация прототипа интернет-платформы для работы с ним интегрирует некоторые хорошо известные тенденции, в том числе:

- изменения подходов к архитектуре программных средств, таких, как графовые представления вычислений, микросервисная архитектура, концепция DevOps;
- развития методов работы с большими данными, таких, как Data-driven approach и Data Mining;
- концентрация усилий на минимизации транзакционных издержек за счет применения интернет-платформ и методов распределенного реестра.

При этом предложенный подход адаптирует эти концепции для любого проекта, который может быть выполнен в цифровом виде.

Предлагаемое нами решение может быть использовано для разработки программного обеспечения для разных отраслей промышленности и государственного управления в рамках развития парадигмы цифровой экономики. В частности, разработанный прототип внедрен в опытно-промышленную эксплуатацию в одной из геофизических компаний России.

Также перспективным представляется разработка на основе описанной цифровой платформы ситуационных центров для органов государственной власти и управления, решающих нетиповые задачи управления опасными и неблагоприятными ситуациями в экономике, социальной и политической сферах. Представлен пример использования предложенного подхода для оперативного поиска региональной информации и ее анализа.

Таким образом, опыт реализации прототипа интернет-платформы, описанной выше, и прикладных проектов с его использованием в различных предметных областях показал принципиальную адекватность и эффективность предлагаемого подхода.

Работа выполнена при частичной финансовой поддержке РФФИ и РФФИ совместно с АН РТ в рамках научных проектов 18-07-00964 и 18-47-160010.

Литература

1. Chesbrough H.W. Open Innovation: The New Imperative for Creating and Profiting from Technology. Cambridge, MA: Harvard Business School Publishing, 2003.
2. Geoffrey G. Parker, Marshall W. Van Alstyne, Sangeet Paul Choudary. Platform Revolution: How Networked Markets are Transforming the Economy and How to Make Them Work for You. W.W.Norton & Company, 2016.
3. Scheer A.-W. Start-Ups are Easy, But... Springer Science and Business Media, 2001.
4. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. URL: https://bitcoin.org/bitcoin.pdf (дата обращения: 23.09.2019).
5. Беляева А.А., Биряльцев Е.В., Галимов М.Р., Демидов Д.Е., Елизаров А.М., Жибрик О.Н. Кластерная архитектура программно-технических средств организации высокопроизводительных систем для нефтегазовой промышленности // Программные системы: теория и приложения. 2017. Т. 8, № 1(32). С. 151–171. DOI: 10.25209/2079-3316-2017-8-1-151-171.
6. Birialtsev E.V., Demidov D.E., Mokshin E.V. Determination of moment tensor and location of microseismic events under conditions of highly correlated noise based on the maximum likelihood method // Geophysical Prospecting. 2017. Vol. 65, no. 6. P. 1510–1526. DOI: 10.1111/1365-2478.12485.
7. Biryal'tsev E.V., Galimov M.R., Elizarov A.M. Software Platform for Mass Supercomputing // Doklady Mathematics. 2017. Vol. 95, no. 2. P. 1–5. DOI: 10.1134/S1064562417020090.
8. Biryal'tsev E.V., Galimov M.R., Elizarov A.M. Workflow-Based Internet Platform for Mass Supercomputing // Lobachevskii Journal of Mathematics. 2018. Vol. 39, no. 5. P. 647–654. DOI: 10.1134/S1995080218050050.
9. Биряльцев Е.В., Минниханов Р.Н. Ситуационный центр главы региона Российской федерации в парадигме цифровой экономики. Современные проблемы безопасности

жизнедеятельности: интеллектуальные транспортные системы и ситуационные центры: материалы V Межд. науч.-практ. конф. / Под общей ред. член-корр. Академии наук Республики Татарстан, д-ра техн. наук, проф. Р.Н. Миниханова. Казань: Центр инновационных технологий, 2018. Ч. II. С. 3–11.

10. Открытый Татарстан. Отчеты ведомств. URL: https://open.tatarstan.ru/reports/categories (дата обращения: 23.09.2019).
11. Elizarov A.M., Lipachev E.K., Nevzorova O.A., Solov'ev V.D. Methods and Means for Semantic Structuring of Electronic Mathematical Documents // Doklady Mathematics. 2014. Vol. 90, no. 1. P. 521–524. DOI: 10.1134/S1064562414050275.
12. Elizarov A.M., Zhizhchenko A.B., Zhil'tsov N.G., Kirillovich A.V., Lipachev E.K. Mathematical knowledge ontologies and recommender systems for collections of documents in physics and mathematics // Doklady Mathematics. 2016. Vol. 93, no. 2. P. 231–233. DOI: 10.1134/S1064562416020174.
13. Elizarov A., Kirillovich A., Lipachev E., Nevzorova O. Digital Ecosystem OntoMath: Mathematical Knowledge Analytics and Management // Communications in Computer and Information Science. Springer, Cham, 2017. Vol. 706. P. 33–46. DOI: 10.1007/978-3-319-57135-5_3.

Биряльцев Евгений Васильевич, к.т.н., заведующий, Центр цифровых технологий, Институт прикладных исследований Академии наук Республики Татарстан; генеральный директор, ООО «Градиент технолоджи» (Казань, Российская Федерация)

Галимов Марат Разифович, к.т.н., заместитель директора по разработке программного обеспечения, ООО «Градиент технолоджи» (Казань, Российская Федерация)

Демидов Денис Евгеньевич, к.ф.-м.н, с.н.с., Казанское отделение, Межведомственный суперкомпьютерный центр Российской академии наук — филиал Федерального государственного учреждения Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук; с.н.с., Высшая школа информационных технологий и интеллектуальных систем Казанского (Приволжского) федерального университета (Казань, Российская Федерация)

Елизаров Александр Михайлович, д.ф.-м.н., профессор, Казанский (Приволжский) федеральный университет; директор, Казанское отделение, Межведомственный суперкомпьютерный центр Российской академии наук — филиал Федерального государственного учреждения Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук (Казань, Российская Федерация)

DOI: 10.14529/cmse200104

DIGITAL PROJECT AND PLATFORM FOR WORKING WITH IT

© 2020 E.V. Biryaltsev[1,2], M.R. Galimov[2], D.E. Demidov[3,4], A.M. Elizarov[3,4]

[1]*Institute of Applied Research of the Academy of Sciences of the Republic of Tatarstan (Levo-Bulachnaya 36A, Kazan, 420111 Russia),*

[2]*Gradient Technology LLC (Bolshaya Krasnaya 63, Kazan, 420111 Russia),*

[3]*Kazan Branch of the Interdepartmental Supercomputer Center of the Russian Academy of Sciences (RAS) — Branch of the Federal State Institution Scientific Center for Research of System Studies of the RAS (Lobachevskogo 2/31, Kazan, 420111 Russia),*

[4]*Kazan (Volga Region) Federal University (Kremlevskaya 35, Kazan, 420111 Russia)*

E-mail: Igenbir@yandex.ru, glmvmrt@gmail.com, dennis.demidov@gmail.com, amelizarov@gmail.com

Received: 09.08.2019

We substantiate the relevance of representing complex scientific digital projects in the form of a directed graph that combines input, output and intermediate data with data processing software into a single digital object. A method for manipulating such a representation in the form of a cloud Internet platform is proposed. The platform includes a central application server and allows to store both algorithms and data, provides means for communication between users, as well as accounting for their use of algorithms and data in solving applied problems. Work with algorithms and data takes place in an executable environment, which is loaded upon joining the platform either onto a user's machine or to a virtual machine in a cloud cluster. This environment allows users to solve applied tasks by creating, modifying and using new algorithms, or reusing the ones from a standard set. User interaction with the server and the repository is carried out through a web interface or a thick client on a local or virtual machine. A working prototype of the named platform is presented, which operates using supercomputer technologies and a desktop virtualization system. The prototype includes tools for creating software based on modular graph architecture and communication services for participants. It is able to execute computational graphs in a high-performance environment, provide registration of intellectual property in the algorithm gallery and carry out billing of its usage. We present examples of using the platform for geophysical research and for public administration.

Keywords: Internet platform, software platform, cloud services, application software, software development, business process automation.

FOR CITATION

Biryaltsev E.V., Galimov M.R., Demidov D.E., Elizarov A.M. Digital Project and Platform for Working with It. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2020. Vol. 9, no. 1. P. 50–68. (in Russian) DOI: 10.14529/cmse200104.

References

1. Chesbrough H.W. Open Innovation: The New Imperative for Creating and Profiting from Technology. Cambridge, MA: Harvard Business School Publishing, 2003.
2. Geoffrey G. Parker, Marshall W. Van Alstyne, Sangeet Paul Choudary. Platform Revolution: How Networked Markets are Transforming the Economy and How to Make Them Work for You. W.W.Norton & Company, 2016.

3. Scheer A.-W. Start-Ups are Easy, But... Springer Science and Business Media, 2001. 220 p.
4. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. Available at: https://bitcoin.org/bitcoin.pdf (accessed: 23.09.2019).
5. Belyaeva A.A., Biryal'cev E.V., Galimov M.R., Demidov D.E., Elizarov A.M., Zhibrik O.N. Architecture of HPC clusters for Oil&Gas Industry. Program systems: Theory and applications. 2017. Vol. 8, no. 1(32). P. 151–171. (in Russian) DOI: 10.25209/2079-3316-2017-8-1-151-171.
6. Birialtsev E.V., Demidov D.E., Mokshin E.V. Determination of moment tensor and location of microseismic events under conditions of highly correlated noise based on the maximum likelihood method. Geophysical Prospecting. 2017. Vol. 65, no. 6. P. 1510–1526. DOI: 10.1111/1365-2478.12485.
7. Biryal'tsev E.V., Galimov M.R., Elizarov A.M. Software Platform for Mass Supercomputing. Doklady Mathematics. 2017. Vol. 95, no. 2. P. 1–5. DOI: 10.1134/S1064562417020090.
8. Biryal'tsev E.V., Galimov M.R., Elizarov A.M. Workflow-Based Internet Platform for Mass Supercomputing. Lobachevskii Journal of Mathematics. 2018. Vol. 39, no. 5. P. 647–654. DOI: 10.1134/S1995080218050050.
9. Biryal'cev E.V., Minnihanov R.N. The situation center of the head of the region of the Russian Federation in the digital economy paradigm. Modern problems of life safety: intelligent transport systems and situational centers: proceedings of the V International Scientific-practical Conference. Kazan, Publishing of the Innovative Technologies Center, 2018. Part II. P. 3–11. (in Russian)
10. Open Tatarstan. Department reports. Available at: https://open.tatarstan.ru/reports/categories (accessed: 23.09.2019). (in Russian)
11. Elizarov A.M., Lipachev E.K., Nevzorova O.A., Solov'ev V.D. Methods and Means for Semantic Structuring of Electronic Mathematical Documents. Doklady Mathematics. 2014. Vol. 90, no. 1. P. 521–524. DOI: 10.1134/S1064562414050275.
12. Elizarov A.M., Zhizhchenko A.B., Zhil'tsov N.G., Kirillovich A.V., Lipachev E.K. Mathematical knowledge ontologies and recommender systems for collections of documents in physics and mathematics. Doklady Mathematics. 2016. Vol. 93, no 2. P. 231–233. DOI: 10.1134/S1064562416020174.
13. Elizarov A., Kirillovich A., Lipachev E., Nevzorova O. Digital Ecosystem OntoMath: Mathematical Knowledge Analytics and Management. Communications in Computer and Information Science. Springer, Cham, 2017. Vol. 706. P. 33–46. DOI: 10.1007/978-3-319-57135-5_3.

УДК 004.021, 519.685 DOI: 10.14529/cmse200105

АНАЛИТИЧЕСКОЕ МОДЕЛИРОВАНИЕ МАТРИЧНО-ВЕКТОРНОГО ПРОИЗВЕДЕНИЯ НА МНОГОЯДЕРНЫХ ПРОЦЕССОРАХ*

© 2020 Е.Н. Акимова[1,2], Р.А. Гареев[1]

[1] *Уральский федеральный университет им. Б.Н. Ельцина (620002 Екатеринбург, ул. Мира, д. 19),*

[2] *Институт математики и механики им. Н.Н. Красовского УрО РАН (620990 Екатеринбург, ул. Софьи Ковалевской, д. 16)*

E-mail: aen15@yandex.ru, gareevroman@gmail.com

Поступила в редакцию: 31.01.2020

Эффективная реализация матрично-векторного произведения имеет существенную практическую значимость в областях машинного обучения, интеллектуального анализа данных, квантовой химии, математической физики, численных методов линейной алгебры, высокопроизводительных вычислений и др. В данной работе представлен алгоритм автоматизированной оптимизации матрично-векторного произведения по времени выполнения, использующийся на этапе компиляции без ручной настройки и автонастройки. Алгоритм основан на моделировании вычислений на гипотетическом многоядерном процессоре, предложенном авторами, с применением полиэдрального представления. В отличие от подходов, основанных на ручной настройке и автонастройке, алгоритм может применяться для создания новых оптимизированных реализаций матрично-векторного произведения в условиях недоступности целевой архитектуры и ограниченности времени выполнения. Алгоритм использован для оптимизации программного кода, реализующего решение структурной обратной задачи гравиметрии о нахождении поверхности раздела сред методом Левенберга—Марквардта. Проведено сравнение производительности полученной реализации с реализациями на основе оптимизированных библиотек линейной алгебры Intel MKL, BLIS, OpenBLAS. Результаты численных экспериментов показывают сравнимость предложенного алгоритма по эффективности с подходами, созданными с использованием ручной настройки при доступе к целевым архитектурам процессоров.

Ключевые слова: компиляторы, линейная алгебра, матрично-векторные операции, аналитическое моделирование, обратная задача гравиметрии.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Акимова Е.Н., Гареев Р.А. Аналитическое моделирование матрично-векторного произведения на многоядерных процессорах // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 1. С. 69–82. DOI: 10.14529/cmse200105.

Введение

Матрично-векторное произведение (matrix-vector multiplication, MVM) и матричное произведение (matrix-matrix multiplication, MMM) широко используются для решения прикладных задач. В частности, MVM вычисляется в процессе обрезки весов (weight pruning) в глубинных нейронных сетях (Deep Neural Networks, DNN), используемых в машинном обучении [1]. MVM применяется в интеллектуальном анализе графов [2]. MVM вычисляется при определении коэффициента скорости реакции в квантовой химии [3]. MVM и MMM используются для решения широкого ряда задач математической физики и, в частности, математической геофизики, численными методами [4]. Минимизация времени выполнения MVM является отдельным предметом изучения высокопроизводительных вычислений [5, 6].

*Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии (ПаВТ) 2020»

Современные подходы, используемые для оптимизации MVM по времени выполнения, основаны на эмпирическом поиске. Их можно разделить на два вида: ручная настройка (manual-tuning) и автонастройка (auto-tuning) [7]. Для выполнения ручной настройки для отдельно рассматриваемой архитектуры компьютера специалист, обладающий знаниями особенностей используемых аппаратных средств, создает новую реализацию требуемой операции. В случае автонастройки для каждой отдельно рассматриваемой компьютерной архитектуры выполняется перебор значений параметров заранее созданной реализации для нахождения их оптимальных значений. Это делает невозможным их применение в процессе кросс-компиляции и в других условиях недоступности целевой архитектуры процессора. Выполнение ручной настройки и автонастройки может потребовать значительных временных затрат. Вследствие этого ручная настройка и автонастройка не могут использоваться, когда время выполнения ограничено. Таким примером являются оптимизации, применяемые компилятором по умолчанию. С целью возможности их частого использования на этапе разработки и компиляции программ предпочтительным являются подходы, требующие наименьшего времени выполнения.

Результаты [7] в области моделирования MMM на гипотетическом процессоре показывают, что эмпирический поиск необязателен для достижения высокой производительности. Рассматриваемые подходы для оптимизации MMM не могут быть использованы для MVM из-за меньшей доли вычислений и большей доли обращений к памяти [8].

В работах [9, 10] представлены алгоритмы оптимизации по времени выполнения обобщенных MMM и сверток тензоров (tensor contraction, TC) в процессе компиляции.

В данной работе предложен алгоритм оптимизации по времени выполнения MVM без эмпирического поиска. Оптимизация выполняется с использованием полиэдрального представления. Для определения значений параметров оптимизации применяется моделирование вычисления MVM на гипотетическом многоядерном процессоре. Алгоритм использован для оптимизации программы, реализующей решение обратной структурной задачи гравиметрии о нахождении поверхности раздела сред методом Левенберга—Марквардта.

Полученный программный код сравнивается по производительности с программными кодами, оптимизированными с помощью библиотек линейной алгебры, созданных с использованием ручной настройки при доступе к целевым архитектурам процессоров (Intel MKL [11], BLIS [12], OpenBLAS [13]). Преимущество предложенного алгоритма состоит в возможности его применения для создания новых оптимизированных реализаций MVM в условиях недоступности целевой архитектуры и ограниченности времени выполнения.

Численные эксперименты проводились на узле с двумя 18-ядерными процессорами Intel Xeon E5-2697 v4 суперкомпьютера «Уран» (Институт математики и механики им. Н.Н. Красовского УрО РАН, Екатеринбург, Россия). Показана сравнимость нашего подхода по производительности с подходами, основанными на эмпирическом поиске.

Статья организована следующим образом. Раздел 1 посвящен описанию алгоритма автоматизированной оптимизации матрично-векторного произведения. В разделе 2 выполнена постановка структурной обратной задачи гравиметрии, описан алгоритм её решения, приведены результаты сравнения производительности полученной реализации решения задачи гравиметрии с реализациями на основе оптимизированных библиотек линейной алгебры. В заключении содержатся выводы и направления дальнейших исследований.

1. Автоматизированная оптимизация MVM

В данном разделе описывается алгоритм автоматизированной оптимизации (ААО) MVM, использующий аналитическое моделирование и оптимизацию полиэдрального представления. Описываемые методы могут использоваться во время кросс-компиляции, выполняемой промышленными компиляторами, и в других случаях, когда время выполнения компиляции ограничено либо архитектура целевого процессора недоступна.

Ручная оптимизация широко применяется для MVM и других операций линейной алгебры, выполняемых экспертами в области ее приложений [5]. Для того чтобы упростить разработку новых высокопроизводительных реализаций под новые архитектуры процессоров, специалистами была предложена автонастройка и аналитическое моделирование [7]. Данные подходы используются для нахождения значений параметров созданных реализаций, обеспечивающих наилучшую производительность программ.

В случае автонастройки время выполнения рассматриваемой реализации измеряется для каждого набора значений параметров. Аналитическое моделирование позволяет смоделировать потребление ресурсов гипотетического процессора (ГП) [7] с целью установления зависимостей между производительностью и значениями параметров рассматриваемой реализации. Для определения наилучших значений параметров реализации MVM в работе предложена модификация аналитической модели, которая использовалась в библиотеке BLIS для MMM.

1.1. Модель гипотетического процессора

ГП описывается следующим образом [7].

- **Архитектура загрузки/сохранения и векторные регистры:** данные должны быть загружены в регистры процессора перед тем как с ними могут быть выполнены вычисления. Имеется N_{REG} векторных регистров. Каждый из них может содержать N_{VEC} значений размера S_{DATA}. Если N_{REG} равно 0, N_{VEC} присваивается значение 1. Предполагается, что команды для работы с памятью могут выполняться одновременно с командами арифметики с плавающей точкой.

- **Векторные инструкции:** пропускная способность процессора составляет N_{VFMA} векторных операций смешанного сложения и умножения (vector fused multiply-add instruction, VFMA) за один такт. Отдельная VFMA выполняет N_{VEC} умножений и сложений, составляющих VFMA. Минимальное количество тактов L_{VFMA}, которое должно быть совершено перед началом выполнения новой зависимой по данным VFMA-инструкции, определяет задержку VFMA-инструкции.

- **Кэш-память:** весь кэш данных — это множественно-ассоциативный кэш с политикой вытеснения последнего по времени использования (Least Recently Used, LRU). Каждый уровень кэша L_i характеризуется следующими параметрами: размер линии кэша C_{L_i}, степень ассоциативности W_{L_i}, количество множеств N_{L_i}, размер S_{L_i}, где $S_{L_i} = N_{L_i} C_{L_i} W_{L_i}$. В случае полностью ассоциативного кэша $N_{\text{L}_i} = 1$.

Авторами предложена модификация ГП, которая определяет инструкции предвыборки, помещающие данные в кэш первого уровня.

- **Инструкции предвыборки:** за один такт процессора может быть выполнено N_{prefetch} инструкций. Задержка каждой инструкции составляет L_{prefetch} тактов процессора. Каждая инструкция может загрузить данные, размер которых равен C_{L_i}.

Помимо инструкций предвыборки модификация определяет минимальное количество тактов L_{VLOAD}, которое должно быть совершено перед началом выполнения новой зависимой по данным векторной инструкции загрузки данных на векторный регистр процессора.

Вычислительная сложность MVM составляет $O(N^2)$, что в N раз меньше, чем вычислительная сложность MMM. Вследствие этого каждый элемент умножаемой матрицы используется только $O(1)$ раз, а не $O(N)$ как в случае MMM. Следовательно, стоимость загрузки элементов матрицы из памяти на регистры процессора преобладает над количеством выполняемых операций. Предвыборка данных, загружающая элементы матрицы в кэш первого уровня до момента их использования, может уменьшить количество промахов кэша и, следовательно, уменьшить стоимость загрузки элементов матрицы. Как следствие, эффективная предвыборка необходима в случае MVM.

1.2. Алгоритмы вычисления MVM

Алгоритм 1 вычисления MVM для случая транспонированной матрицы имеет следующий вид:

Алгоритм 1: Вычисление MVM. Случай транспонированной матрицы

```
1  begin
2      for j_c = 0...N step N_c do
3          for i_c = 0...M step M_c do
4              for j_b = 0...N_c step N_b do
5                  Выполняем предвыборку M_c × N_b элементов матрицы A с шагом D
6                  for i_b = 0...M_c step 1 do
7                      I = i_c + i_b
8                      acc(0:N_r − 1) = x(I)
9                      for j_r = 0...N_b step N_r do
10                         𝕁 = j_c + j_b + j_r : j_c + j_b + j_r + N_r − 1
11                         y(𝕁) += A(I, 𝕁) · acc(0 : N_r − 1)
12                     end
13                 end
14             end
15         end
16     end
17 end
```

Алгоритм 2 вычисления MVM для случая нетранспонированной матрицы имеет следующий вид:

Алгоритм 2: Вычисление MVM. Случай нетранспонированной матрицы

```
1  begin
2      for j_c = 0...N step N_c do
3          for i_c = 0...M step M_c do
4              for j_r = 0...N_c step N_r do
5                  if j_r mod 4C_L1/S_DATA == 0 then
6                      Предвыборка M_c × 4C_L1/S_DATA элементов A с шагом D
7                      Предвыборка 4C_L1/S_DATA элементов x с шагом D
8                  end
9              end
10             J = j_c + j_r : j_c + j_r + N_r − 1
11             acc(0, 0 : N_r − 1) += A(i_c, J) · x(J)
12             ...
13             acc(M_c − 1, 0 : N_r − 1) += A(i_c + M_c − 1, J) · x(J)
14         end
15         y(i_c) += Σ_{i=0}^{N_r−1} acc(0, i)
16         ...
17         y(i_c + M_c − 1) += Σ_{i=0}^{N_r−1} acc(M_c − 1, i)
18     end
19 end
```

В процессе выполнения алгоритмов 1 и 2 матрицы и векторы разбиваются на блоки. Это влияет на выполняемую предвыборку данных, загружающую элементы матрицы в кэш первого уровня, а также позволяет использовать элементы блоков, хранящихся в кэш-памяти, повторно. Размеры блоков и размер шага предвыборки являются параметрами алгоритма. Их оптимальные значения определяются с помощью гипотетического процессора, описанного в разделе 1.1.

В случае транспонированной матрицы формулы для нахождения значений параметров имеют следующий вид:

$$N_b = \min\left\{ \max\left\{N_{\text{VFMA}} L_{\text{VFMA}} N_{\text{VEC}}, (N_{\text{REG}} - 2) N_{\text{VEC}}\right\}, \left\lceil \frac{N_{L_1} C_{L_1}}{N_{\text{VEC}} S_{\text{DATA}}} \right\rceil N_{\text{VEC}} \right\},$$

$$N_r = N_{\text{VEC}}, \quad M_c = \min\left\{W_{L_2} - 1, W_{L_1}\right\}, \quad N_c = N_{L_2} C_{L_2} / S_{\text{DATA}},$$

$$D = \left\lceil \frac{L_{\text{prefetch}}}{\lceil M_c \lceil N_b S_{\text{DATA}} / C_{L_1} \rceil / N_{\text{prefetch}} \rceil + M_c \left(\lceil N_b / (N_{\text{VEC}} N_{\text{VFMA}}) \rceil + L_{\text{VLOAD}} \right)} \right\rceil.$$

В случае нетранспонированной матрицы формулы для нахождения значений параметров имеют следующий вид:

$$N_c = N_{L_2} C_{L_2} / S_{\text{DATA}}, \quad N_r = N_{\text{VEC}} \left\lceil \gamma / \min\left\{W_{L_1}, W_{L_2} - 1\right\} \right\rceil, \quad M_c = \lceil \gamma N_{\text{VEC}} / N_r \rceil,$$
$$\gamma = N_{\text{VFMA}} L_{\text{VFMA}},$$

$$D = \left\lceil \frac{L_{\text{prefetch}}}{\lceil 4(M_c+1)/N_{\text{prefetch}} \rceil + 4C_{L_1}\left(\lceil M_c N_r / (N_{\text{VFMA}} N_{\text{VEC}}) \rceil + L_{\text{VLOAD}} \right) / (S_{\text{DATA}} N_r)} \right\rceil .$$

Значение параметров N_{REG}, N_{VEC}, S_{DATA}, N_{VFMA}, L_{VFMA}, N_{L_i}, C_{L_i}, W_{L_i}, L_{prefetch}, N_{prefetch}, L_{VLOAD} являются параметрами модели гипотетического процессора, описанной в разделе 1.1.

В реальных случаях параметры определяются техническими характеристиками целевого процессора.

1.3. Комплекс программ

Для автоматизированного распознавания и оптимизации MVM в промышленных компиляторах применяется полиэдральная компиляция. Основой такого подхода служит полиэдральное представление программы [14].

Определение. Полиэдральная модель — это математическое представление (фреймворк) для моделирования и оптимизации доступа к памяти, производимого в группах вложенных циклов, не рассматривая отдельные вычисления.

Инфраструктура для полиэдральной компиляции выполняет построение и оптимизацию полиэдрального представления программы, используя промежуточное представление программного кода, генерируемого компилятором. Промежуточное представление создается фронтендом компилятора с целью дальнейшей оптимизации и генерации кода для выбранной архитектуры [15].

В работах [9, 10] авторами показано, что аналитическое моделирование MMM, выполняемое BLIS, и автоматизированная оптимизация и распараллеливание, выполняемые над полиэдральным представлением, могут быть использованы для достижения производительности реализаций, созданных для случая свертки тензоров TC. Отметим, что MVM является важным частным случаем TC.

На рис. 1 изображен комплекс автоматизированной оптимизации обобщенных тензорных операций (AOOTO), представленный в работах [9, 10]. Комплекс AOOTO имеет три основные части: фронтенд, выполняющий компиляцию программы и построение промежуточного представления; инфраструктуру для построения полиэдрального представления и его модификаций, например, для распознавания TC и ее оптимизации (на рис. 1 выделено цветом); бэкенд, выполняющий оптимизацию промежуточного кода программы, а также генерацию ассемблерного кода для целевой архитектуры. Такая структура позволяет обеспечить тестируемость описанных частей и их взаимозаменяемость.

В качестве фронтенда, выполняющего компиляцию в комплексе AOOTO, использован фронтенд Clang [16]. Комплекс оптимизаций Polly [15] применен в качестве инфраструктуры для построения полиэдрального представления, а также в качестве основы для реализации распознавания TC и его оптимизации. Внешняя библиотека LLVM Core [16] использована в качестве бэкенда, выполняющего оптимизацию промежуточного представления программы, а также генерацию ассемблерного кода для целевой архитектуры.

Существует множество подходов для оптимизации TC по времени выполнения. В частности, реализация TC может быть оптимизирована, используя последовательности оптимизаций циклов [17]. Для оптимизации TC могут быть использованы высокопроизводительные готовые реализации. Фреймворки GETT (The general matrix multiplication (GEMM)-like

Tensor-Tensor multiplication) [18] и TBLIS (Tensor-Based Library Instantiation Software) [19] используют микро-ядра МММ. Микро-ядра являются частями готовой реализации МММ, содержащие векторные инструкции. Фреймворк Polly использует микро-ядра МММ и генерирует их автоматизировано.

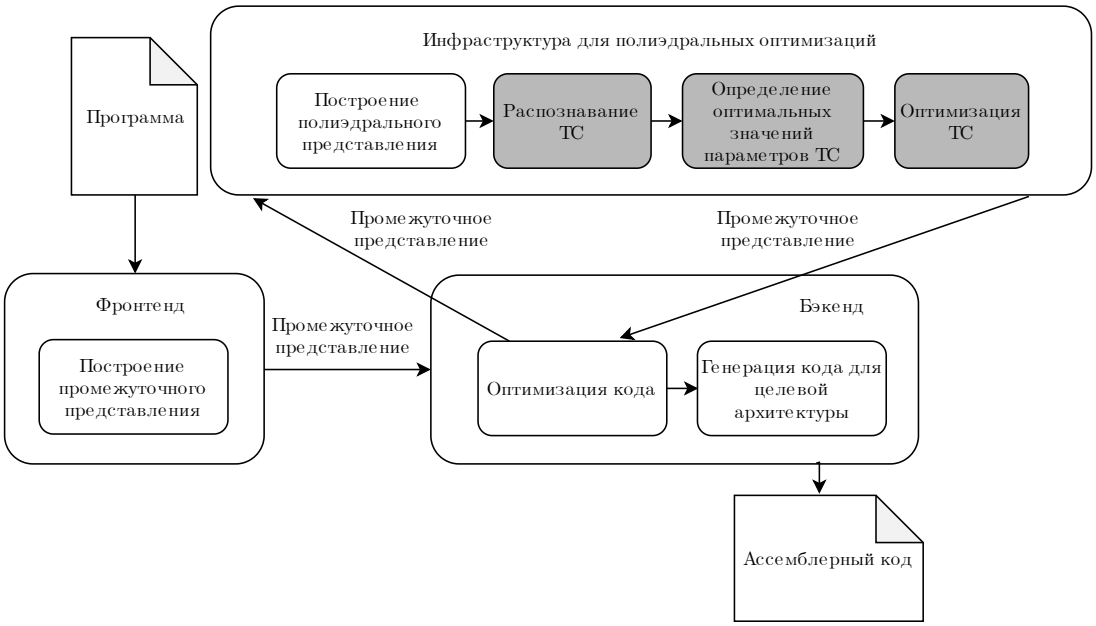


Рис. 1. Комплекс программ

Polly не поддерживает оптимизацию MVM. В ближайшее время предполагается внедрение описанного алгоритма ААО MVM в Polly. Новые инструкции полиэдрального представления могут быть сгенерированы для выполнения предвыборки данных в кэш первого уровня выбранных итераций циклов. Для нетранспонированных матриц в MVM нужно сгенерировать и векторизовать редукцию элементов вычисляемого вектора [5, 20].

2. Численные эксперименты

2.1. Постановка обратной задачи гравиметрии и метод решения

Рассмотрим трехмерную структурную обратную задачу гравиметрии о восстановлении поверхности раздела между средами.

Задача состоит в нахождении функции $\zeta = \zeta(x, y)$, описывающей поверхность раздела сред постоянной плотности σ_1 и σ_2 по гравитационному полю $\Delta g(x, y, 0)$, измеренному на некоторой площади земной поверхности, и скачку плотности на границе раздела $\Delta\sigma = \sigma_1 - \sigma_2$. Пусть $z = h$ — асимптотическая плоскость для данной поверхности раздела ζ такая, что $\lim\limits_{x,y\to\pm\infty} \zeta(x, y) = h$.

Задача описывается двумерным нелинейным интегральным уравнением первого рода [21]:

$$f\Delta\sigma \int\limits_{-\infty}^{\infty}\int\limits_{-\infty}^{\infty} \left\{ \frac{1}{\sqrt{((x-x')^2 + (y-y')^2 + \zeta^2(x', y'))}} - \right.$$
$$\left. - \frac{1}{\sqrt{((x-x')^2 + (y-y')^2 + h^2)}} \right\} dx'dy' = \Delta g(x, y, 0), \tag{1}$$

где f — гравитационная постоянная.

Предположим, что значения поля вне некоторой прямоугольной области $\Pi = \{(x, y) : a \leqslant x \leqslant b, c \leqslant y \leqslant d\}$ близки к нулю. Проведем дискретизацию области Π на сетке $N \times M$: $(x_i, y_j), \quad i = 1, ..., N, \quad j = 1, \ldots, M$ с шагами Δx и Δy.

После аппроксимации интегрального оператора по квадратурным формулам получим уравнение

$$\begin{aligned} & f\Delta\sigma\Delta x\Delta y\times \\ & \times\sum_{i=1}^{N}\sum_{j=1}^{M}\left[\frac{1}{\sqrt{(x_i-x_u)^2+(y_j-y_v)^2+\zeta^2(x_i,y_j)}}-\right. \\ & \left.-\frac{1}{\sqrt{(x_i-x_u)^2+(y_j-y_v)^2+h^2}}\right]=\Delta g(x_u,y_v,0), \\ & u=1,...,M, \quad v=1,...,N. \end{aligned} \tag{1a}$$

Правую часть $\Delta g(x_u, y_v, 0)$ представим в виде вектора

$$\begin{aligned} & F = \{F_{(v-1)M+u}\} = \{\Delta g(x_u, y_v, 0)\}, \\ & u = 1, ..., M, \quad v = 1, ..., N \end{aligned}$$

и искомую поверхность представим в виде вектора

$$z = \{z_{(j-1)M+i}\} = \{\zeta(x_i, y_j), \quad i = 1, ..., M, \quad j = 1, ..., N\}.$$

Тогда систему (1a) запишем в операторном виде

$$A(z) = F. \tag{2}$$

Для решения системы (2) будем использовать метод Левенберга—Марквардта [22]

$$z^{k+1} = z^k - \gamma\left[\left(A'(z^k)\right)^{\intercal} A'(z^k) + \alpha I\right]^{-1} \times A'(z^k)^{\intercal}(A'(z^k) - F), \tag{3}$$

где z^k — приближенное решение на k-ой итерации, $A'(z^k)$ — производная Фреше оператора A в точке u^k, I — единичный оператор, γ — демпфирующий множитель, α — параметр регуляризации.

В качестве начального приближения используется $z^0 \equiv h$. Критерием останова является условие $\left\|A(z^k) - F\right\| / \|F\| \leqslant \varepsilon_1$ при некотором $\varepsilon_1 < 0$.

Вместо выполнения трудоемкой процедуры обращения матрицы воспользуемся следующим приемом. На каждом шаге итерационного метода Левенберга—Марквардта [22] решаем систему линейных алгебраических уравнений

$$Bz = b, \tag{4}$$

где

$$B = \left(A'(z^k)\right)^{\intercal} A'(z^k) + \alpha I, \quad b = \left[\left(A'(z^k)\right)^{\intercal} A'(z^k) + \alpha I\right] z^k - \gamma A'(z^k)^{\intercal}(A'(z^k) - F).$$

Для решения (4) используется метод минимальных невязок [22]

$$z^{l+1} = z^l - \frac{\langle B(Bz^l - b), Bz^l - b\rangle}{\|B(Bz^l - b)\|^2}(Bz^l - b), \tag{5}$$

где z^l — приближенное решение на l-ой итерации метода минимальных невязок.

В качестве начального приближения используется $z^0 \equiv 0$. Критерием останова является условие $\left\|Bz^l - b\right\|/\|b\| \leqslant \varepsilon_2$ при некотором $\varepsilon_2 < 0$.

Замечание. Метод Левенберга—Марквардта (3) и метод минимальных невязок (5) на каждом шаге итерационного процесса (3) требуют выполнения значительного количества операций MVM. Поэтому предложенный алгоритм автоматической оптимизации MVM может быть использован для автоматизированной и эффективной оптимизации соответствующей программы.

2.2. Результаты экспериментов

В данном разделе проведено сравнение производительности нескольких реализаций программного кода для решения обратной задачи гравиметрии (раздел 2.1) с использованием библиотек линейной алгебры Intel MKL, BLIS, OpenBLAS с реализацией, полученной с использованием предлагаемого алгоритма AAO MVM (см. раздел 1).

Для выполнения экспериментов использовались два 18-ядерных процессора Intel Xeon E5-2697 v4 суперкомпьютера «Уран». Для распараллеливания программы применялись директивы стандарта OpenMP и группы из 36 потоков.

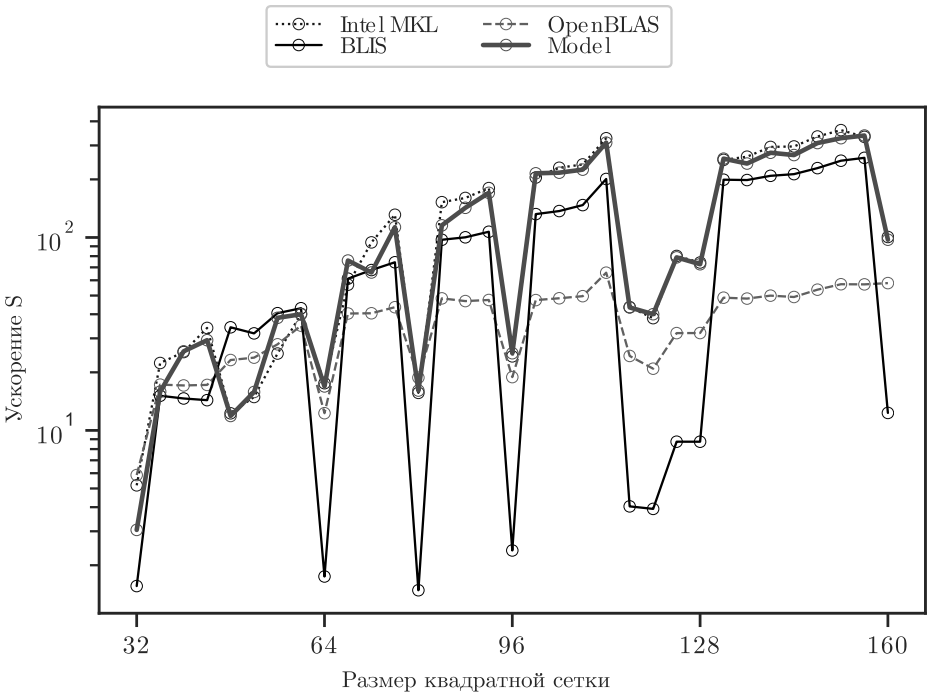


Рис. 2. Ускорение, полученное для обратной задачи гравиметрии

Определим ускорение $S = T_{serial}/T_{parallel}$, где T_{serial} – время выполнения последовательного кода программы с применением стандартных оптимизаций компилятора третьего уровня (O3), $T_{parallel}$ – время выполнения параллельного кода программы с использованием библиотечных процедур либо кода, оптимизированного с помощью предлагаемого алгоритма.

На рис. 2 для 36 ядер процессора Intel Xeon E5-2697 v4 суперкомпьютера «Уран» изображены коэффициенты ускорения для решения обратной задачи гравиметрии методом Левенберга—Марквардта на квадратных сетках с размерами, изменяющимися от 32 до

160 с шагом 4. Представленные коэффициенты ускорения получены для разработанного алгоритма AAO MVM, а также для библиотек BLIS, OpenBLAS и Intel MKL.

В большинстве случаев наилучшие результаты получены с использованием библиотеки Intel MKL. Результаты алгоритма AAO MVM сравнимы с библиотечными для всех рассмотренных размеров сеток. В среднем они отличаются не более чем на 1% и превосходят результаты библиотек OpenBLAS и BLIS.

На основании результатов численных экспериментов можно сделать вывод, что реализация программы, использующей описанный метод оптимизации MVM, сравнима по производительности с реализациями, использующими библиотеки BLIS, OpenBLAS и Intel MKL, оптимизированные вручную. Результаты численных экспериментов показывают, что значения параметров реализации, полученные с использованием аналитического моделирования (раздел 1), не зависят от размерности задачи.

Заключение

В данной работе представлен алгоритм автоматизированной оптимизации матрично-векторного произведения по времени выполнения, использующийся на этапе компиляции. Алгоритм AAO MVM основан на оптимизации полиэдрального представления программы и моделировании вычислений на гипотетическом многоядерном процессоре. В отличие от подходов, основанных на ручной настройке и автонастройке, описанный алгоритм может применяться для создания новых реализаций матрично-векторного произведения в условиях недоступности целевой архитектуры процессора и ограниченности времени выполнения.

Алгоритм использован для оптимизации программного кода, реализующего решение структурной обратной задачи гравиметрии о нахождении поверхности раздела сред методом Левенберга—Марквардта. Проведено сравнение производительности полученной реализации с реализациями на основе оптимизированных библиотек линейной алгебры Intel MKL, BLIS, OpenBLAS. Результаты численных экспериментов показывают сравнимость предложенного алгоритма по эффективности с подходами, созданными с использованием эмпирического поиска при доступе к целевым архитектурам процессоров. В дальнейшем планируется внедрение описанного алгоритма в комплекс оптимизаций Polly с целью использования в процессе компиляции, выполняемой фронтендом Clang.

Перспективным направлением будущих исследований является применение аналитического моделирования для оптимизации матрично-векторного произведения матриц специального вида (матрица Теплица и др.). В основе таких подходов может использоваться симметрия элементов и другие свойства матриц.

Литература

1. Yu J., Lukefahr A., Palframan D., *et al.* Scalpel: Customizing DNN Pruning to the Underlying Hardware Parallelism // SIGARCH Computer Architecture News. 2017. Vol. 45, no. 2, P. 548–560. DOI: 10.1145/3140659.3080215.

2. Yang X., Parthasarathy S., Sadayappan P. Fast Sparse Matrix-vector Multiplication on GPUs: Implications for Graph Mining // Proceedings of the VLDB Endowment. 2011. Vol. 4, no. 4. P. 231–242. DOI: 10.14778/1938545.1938548.

3. Kaushik D., Gropp W., Minkoff M., *et al.* Improving the Performance of Tensor Matrix Vector Multiplication in Cumulative Reaction Probability Based Quantum Chemistry Codes // High Performance Computing (HiPC 2008). Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

P. 120–130. DOI: 10.2172/928654.

4. Martyshko P.S., Akimova E.N., Misilov V.E. Solving the structural inverse gravity problem by the modified gradient methods // Izvestiya, Physics of the Solid Earth. 2016. Vol. 52, no. 5. P. 704–708. DOI: 10.1134/S1069351316050098.
5. Hassan S.A., Mahmoud M.M., Hemeida A., *et al.* Effective Implementation of Matrix-Vector Multiplication on Intel's AVX Multicore Processor // Computer Languages, Systems & Structures. 2018. Vol. 51, P. 158–175. DOI: 10.1016/j.cl.2017.06.003.
6. Liang J., Zhang Y. Optimization of GEMV on Intel AVX processor // International Journal of Database Theory and Application. 2016. Vol. 9. P. 47–60. DOI: 10.14257/ijdta.2016.9.2.06.
7. Low T.M., Igual F.D., Smith T.M. Quintana-Orti E.S., Analytical Modeling Is Enough for High-Performance BLIS // ACM Transactions on Mathematical Software. 2016. Vol. 43, no. 2. P. 12:1–12:18. DOI: 10.1145/2925987.
8. Frison G. Algorithms and Methods for High-Performance Model Predictive Control // Technical University of Denmark. 2016. 345 c.
9. Akimova E.N., Gareev R.A. Algorithm of Automatic Parallelization of Generalized Matrix Multiplication // CEUR Workshop Proceedings. 2017. Vol. 2005. P. 1–10.
10. Gareev R., Grosser T., Kruse M. High-Performance Generalized Tensor Operations: A Compiler-Oriented Approach // ACM Transactions on Architecture and Code Optimization. 2018. Vol. 15, no. 3. P. 34:1–34:27. DOI: 10.1145/3235029.
11. Intel. Intel Math Kernel Library (Intel MKL). URL: `https://software.intel.com/en-us/mkl` (дата обращения: 27.10.2019).
12. Van Zee F.G., Van De Geijn R.A. BLIS: A Framework for Rapidly Instantiating BLAS Functionality // ACM Transactions on Mathematical Software. 2015. Vol. 41, no. 3. P. 14:1–14:33. DOI: 10.1145/2764454.
13. Xianyi Z., Qian W., Yunquan Z. Model-driven level 3 BLAS performance optimization on Loongson 3A processor // Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on Parallel and Distributed Systems, IEEE. 2012. P. 684–691. DOI: 10.1109/icpads.2012.97.
14. Feautrier P., Lengauer C. Polyhedron Model // Encyclopedia of Parallel Computing. Springer US, Boston, MA. 2011. P. 1581–1592. DOI: 10.1007/978-0-387-09766-4_502.
15. Grosser T., Größlinger A., Lengauer C. Polly—Performing polyhedral optimizations on a low-level intermediate representation // Parallel Process. Lett. 2012. Vol. 22. no. 4. DOI: 10.1142/S0129626412500107.
16. Lattner C. LLVM: An Infrastructure for Multi-Stage Optimization // Master's Thesis. 2002. URL: `http://llvm.cs.uiuc.edu` (дата обращения: 27.10.2019).
17. Apra E., Klemm M., Kowalski K. Efficient Implementation of Many-Body Quantum Chemical Methods on the Intel®Xeon Phi™Coprocessor // International Conference for High Performance Computing, Networking, Storage and Analysis. 2014. P. 674–684. DOI: 10.1109/SC.2014.60
18. Springer P., Bientinesi P. Design of a high-performance GEMM-like tensor-tensor multiplication // ACM Trans. Math. Softw. 2018. Vol. 44, no. 3, Article 28. DOI: 10.1145/3157733.

19. Matthews D. High-performance tensor contraction without BLAS // SIAM Journal on Scientific Computing. 2016. Vol. 40. DOI: 10.1137/16M108968X

20. Doerfert J., Streit K., Hack S., *et al.* Polly's polyhedral scheduling in the presence of reductions // arXiv preprint. 2015. arXiv:1505.07716.

21. Numerov B.V. Interpretation of gravitational observations in the case of one contact surface // Dokl. Akad. Nauk SSSR. 1930. P. 569–574.

22. Vasin V.V., Eremin I.I. Operators and iterative processes of Fejér type: theory and applications // Walter de Gruyter. 2009. Vol. 53. 155 p. DOI: 10.1515/9783110218190.

Акимова Елена Николаевна, д.ф.-м.н., доцент, ведущий научный сотрудник Института математики и механики им. Н.Н. Красовского Уральского отделения РАН (Екатеринбург, Российская Федерация); профессор кафедры информационных технологий и систем управления, Институт радиоэлектроники и информационных технологий-РтФ, УрФУ имени первого Президента России Б.Н. Ельцина (Екатеринбург, Российская Федерация)

Гареев Роман Альбертович, аспирант, Институт радиоэлектроники и информационных технологий-РтФ, УрФУ имени первого Президента России Б.Н. Ельцина (Екатеринбург, Российская Федерация)

DOI: 10.14529/cmse200105

APPLICATION OF ANALYTICAL MODELING OF MATRIX-VECTOR MULTIPLICATION ON MULTICORE PROCESSORS

© 2020 E.N. Akimova[1,2], R.A. Gareev[1]

[1] *Ural Federal University*
(Mira 19, Yekaterinburg, 620002 Russia),
[2] *N.N. Krasovskii Institute of Mathematic sand Mechanics, UrB RAS*
(S. Kovalevskaya 16, Yekaterinburg, 620990 Russia)
E-mail: aen15@yandex.ru, gareevroman@gmail.com
Received: 31.01.2020

Many areas of study and their applications such as machine learning, data mining, quantum chemistry, mathematical physics, and high-performance computing require effective implementation of matrix-vector multiplication. In this work, we present an overview of the algorithm for automatic optimization of matrix-vector multiplication. The algorithm models computations on the hypothetical multicore processor, which is introduced by the authors, and applies polyhedral modeling. In comparison with methods, which rely on manual tuning and auto-tuning, the algorithm can be utilized when the execution time is a critical factor and the target platform is not accessible. We apply the approach to optimize an implementation of the solution of the inverse gravimetry problem of finding an interface between the layers, which uses the iterative Levenberg—Marquardt method. The performance of the obtained implementation is compared with the performances produced by implementations, which are based on MKL, BLIS, and OpenBLAS. Results of the experimental evaluation show that the considered approach is comparable with the approaches, which are created on target architectures using manual tuning.

Keywords: compilers, linear algebra, matrix-vector operations, analytical modeling, inverse gravimetry problem.

FOR CITATION

Akimova E.N., Gareev R.A. Application of Analytical Modeling of Matrix-Vector Multiplication on Multicore Processors. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2020. Vol. 9, no. 1. P. 69–82. (in Russian) DOI: 10.14529/cmse200105.

References

1. Yu J., Lukefahr A., Palframan D., *et al.* Scalpel: Customizing DNN Pruning to the Underlying Hardware Parallelism. SIGARCH Computer Architecture News. 2017. Vol. 45, no. 2. P. 548–560. DOI: 10.1145/3140659.3080215.

2. Yang X., Parthasarathy S., Sadayappan P. Fast Sparse Matrix-vector Multiplication on GPUs: Implications for Graph Mining. Proceedings of the VLDB Endowment. 2011. Vol. 4, no. 4. P. 231–242. DOI: 10.14778/1938545.1938548.

3. Kaushik D., Gropp W., Minkoff M., *et al.* Improving the Performance of Tensor Matrix Vector Multiplication in Cumulative Reaction Probability Based Quantum Chemistry Codes. High Performance Computing (HiPC 2008). Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. P. 120–130. DOI: 10.2172/928654.

4. Martyshko P.S., Akimova E.N., Misilov V.E. Solving the structural inverse gravity problem by the modified gradient methods. Izvestiya, Physics of the Solid Earth. 2016. Vol. 52, no. 5. P. 704–708. DOI: 10.1134/S1069351316050098.

5. Hassan S.A., Mahmoud M.M., Hemeida A., *et al.* Effective Implementation of Matrix-Vector Multiplication on Intel's AVX Multicore Processor. Computer Languages, Systems & Structures. 2018. Vol. 51. P. 158–175. DOI: 10.1016/j.cl.2017.06.003.

6. Liang J., Zhang Y. Optimization of GEMV on Intel AVX processor. International Journal of Database Theory and Application. 2016. Vol. 9. P. 47–60. DOI: 10.14257/ijdta.2016.9.2.06.

7. Low T.M., Igual F.D., Smith T.M. Quintana-Orti E.S., Analytical Modeling Is Enough for High-Performance BLIS. ACM Transactions on Mathematical Software. 2016. Vol. 43, no. 2. P. 12:1–12:18. DOI: 10.1145/2925987.

8. Frison G. Algorithms and Methods for High-Performance Model Predictive Control. Technical University of Denmark. 2016. 345 p.

9. Akimova E.N., Gareev R.A. Algorithm of Automatic Parallelization of Generalized Matrix Multiplication. CEUR Workshop Proceedings. 2017. Vol. 2005. P. 1–10.

10. Gareev R., Grosser T., Kruse M. High-Performance Generalized Tensor Operations: A Compiler-Oriented Approach. ACM Transactions on Architecture and Code Optimization. 2018. Vol. 15, no. 3. P. 34:1–34:27. DOI: 10.1145/3235029.

11. Intel. Intel Math Kernel Library (Intel MKL). Available at: `https://software.intel.com/en-us/mkl` (accessed: 27.10.2019).

12. Van Zee F.G., Van De Geijn R.A. BLIS: A Framework for Rapidly Instantiating BLAS Functionality. ACM Transactions on Mathematical Software. 2015. Vol. 41, no. 3. P. 14:1–14:33. DOI: 10.1145/2764454.

13. Xianyi Z., Qian W., Yunquan Z. Model-driven level 3 BLAS performance optimization on Loongson 3A processor. Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on Parallel and Distributed Systems, IEEE. 2012. P. 684–691. DOI: 10.1109/icpads.2012.97.

14. Feautrier P., Lengauer C. Polyhedron Model. Encyclopedia of Parallel Computing. Springer US, Boston, MA. 2011. P. 1581–1592. DOI: 10.1007/978-0 387-09766-4_502.

15. Grosser T., Größlinger A., Lengauer C. Polly—Performing polyhedral optimizations on a low-level intermediate representation. Parallel Process. Lett. 2012. Vol. 22, no. 4. DOI: 10.1142/S0129626412500107.

16. Lattner C. LLVM: An Infrastructure for Multi-Stage Optimization. Master's Thesis. 2002. Available at: `http://llvm.cs.uiuc.edu` (accessed: 27.10.2019).

17. Apra E., Klemm M., Kowalski K. Efficient Implementation of Many-Body Quantum Chemical Methods on the Intel®Xeon Phi™Coprocessor. International Conference for High Performance Computing, Networking, Storage and Analysis. 2014. P. 674–684. DOI: 10.1109/SC.2014.60

18. Springer P., Bientinesi P. Design of a high-performance GEMM-like tensor-tensor multiplication. ACM Trans. Math. Softw. 2018. Vol. 44, no. 3, Article 28. DOI: 10.1145/3157733.

19. Matthews D. High-performance tensor contraction without BLAS. SIAM Journal on Scientific Computing. 2016. Vol. 40. DOI: 10.1137/16M108968X.

20. Doerfert J., Streit K., Hack S., *et al.* Polly's polyhedral scheduling in the presence of reductions. arXiv preprint. 2015. arXiv:1505.07716.

21. Numerov B.V. Interpretation of gravitational observations in the case of one contact surface. Dokl. Akad. Nauk SSSR. 1930. P. 569–574.

22. Vasin V.V., Eremin I.I. Operators and iterative processes of Fejér type: theory and applications. Walter de Gruyter. 2009. Vol. 53. 155 p. DOI: 10.1515/9783110218190.

СВЕДЕНИЯ ОБ ИЗДАНИИ

Научный журнал «Вестник ЮУрГУ. Серия «Вычислительная математика и информатика» основан в 2012 году.

Учредитель — Федеральное государственное автономное образовательное учреждение высшего образования «Южно-Уральский государственный университет» (национальный исследовательский университет).

Главный редактор — Л.Б. Соколинский.

Свидетельство о регистрации ПИ ФС77-57377 выдано 24 марта 2014 г. Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций.

Журнал включен в Реферативный журнал и Базы данных ВИНИТИ; индексируется в библиографической базе данных РИНЦ. Журнал размещен в открытом доступе на Всероссийском математическом портале MathNet. Сведения о журнале ежегодно публикуются в международной справочной системе по периодическим и продолжающимся изданиям «Ulrich's Periodicals Directory».

Решением Президиума Высшей аттестационной комиссии Министерства образования и науки Российской Федерации журнал включен в «Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты на соискание ученой степени кандидата наук, на соискание ученой степени доктора наук» по научным специальностям и соответствующим им отраслям науки: 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей (физико-математические науки), 05.13.17 – Теоретические основы информатики (физико-математические науки).

Подписной индекс научного журнала «Вестник ЮУрГУ», серия «Вычислительная математика и информатика»: 10244, каталог «Пресса России». Периодичность выхода — 4 выпуска в год.

Адрес редакции, издателя: 454080, г. Челябинск, проспект Ленина, 76, Издательский центр ЮУрГУ, каб. 32.

ПРАВИЛА ДЛЯ АВТОРОВ

1. Правила подготовки рукописей и пример оформления статей можно загрузить с сайта серии `http://vestnikvmi.susu.ru`. **Статьи, оформленные без соблюдения правил, к рассмотрению не принимаются.**
2. Адрес редакционной коллегии научного журнала «Вестник ЮУрГУ», серия «Вычислительная математика и информатика»:

   Россия 454080, г. Челябинск, пр. им. В.И. Ленина, 76, ЮУрГУ, кафедра СП,

   ответственному секретарю Цымблеру М.Л.
3. Адрес электронной почты редакции: `vestnikvmi@susu.ru`
4. **Плата с авторов за публикацию рукописей не взимается, и гонорары авторам не выплачиваются.**

ВЕСТНИК
ЮЖНО-УРАЛЬСКОГО
ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
Серия
«ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА»
Том 9, № 1
2020

16+

Техн. редактор *А.В. Миних*

Издательский центр Южно-Уральского государственного университета

Подписано в печать 24.02.2020. Дата выхода в свет 02.03.2020. Формат 60×84 1/8. Печать цифровая.
Усл. печ. л. 9,76. Тираж 500 экз. Заказ 53/116. Цена свободная.

Отпечатано в типографии Издательского центра ЮУрГУ.
454080, г. Челябинск, проспект Ленина, 76.