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

2021, том 10, № 3

SERIES

COMPUTATIONAL MATHEMATICS AND SOFTWARE ENGINEERING

2021, volume 10, no. 3

1943

ВЕСТНИК

ЮЖНО-УРАЛЬСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2021
Т. 10, № 3

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"

2021
Vol. 10, no. 3

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 519.64, 517.922 **DOI: 10.14529/cmse210301**

ОБ ОДНОМ МЕТОДЕ ЧИСЛЕННОГО РЕШЕНИЯ ВЫРОЖДЕННЫХ ИНТЕГРО-ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ СО СЛАБОЙ ОСОБЕННОСТЬЮ В ЯДРЕ

© 2021 Е.В. Чистякова[1], Л.С. Соловарова[1], Доан Тай Сон[2]

[1] *Институт динамики систем и теории управления им. В.М. Матросова СО РАН*
(664033 Иркутск, ул. Лермонтова, д. 134)

[2] *Институт математики Вьетнамской академии наук и технологий*
(10307 Вьетнам, Ханой, ул. Хоанг куок вьет роуд, д. 18)

E-mail: elena.chistyakova@icc.ru, soleilu@mail.ru, dtson@math.ac.vn

Поступила в редакцию: 02.03.2021

Формулировки многих прикладных задач часто включают в себя дифференциальные уравнения и интегральные уравнения Вольтерра первого и второго рода. Комбинируя такие уравнения, мы получаем систему интегро-дифференциальных уравнений с вырожденной матрицей перед главной частью. Такие системы называются вырожденными интегро-дифференциальными уравнениями. Если они не содержат интегральную составляющую, то их называют дифференциально-алгебраическими уравнениями. Если отсутствует слагаемое с производной, то их принято называть интегро-алгебраическими уравнениями. К подобным математическим формулировкам приводит моделирование процессов, протекающих в электрических и гидравлических цепях, различных динамических системах, в частности, многотельных. Поэтому качественное исследование и численное решение такого рода задач являются достаточно актуальными, а результаты исследований — востребованными на практике. В данной статье на основе теории матричных пучков, а также с использованием схем исследований, разработанных для дифференциально-алгебраических и интегро-алгебраических уравнений, проанализированы условия существования и единственности решения вырожденных интегро-дифференциальных уравнений со слабой особенностью в ядре и предложен численный метод их решения, который был реализован в пакете прикладных программ MATLAB и протестирован на модельных примерах.

Ключевые слова: дифференциальные уравнения, интегро-дифференциальные уравнения, уравнение Абеля, слабая особенность.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Чистякова Е.В., Соловарова Л.С., Доан Тай Сон. Об одном методе численного решения вырожденных интегро-дифференциальных уравнений со слабой особенностью в ядре // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 3. С. 5–15. DOI: 10.14529/cmse210301.

Введение

Формулировки многих прикладных задач могут включать в себя дифференциальные уравнения и интегральные уравнения Вольтерра первого и второго рода, а также алгебраические связи. Комбинируя такие уравнения, мы получаем систему интегро-дифференциальных уравнений (ИДУ) с вырожденной матрицей перед главной частью. Такие системы называются вырожденными интегро-дифференциальными уравнениями. Если они не содержат интегральную составляющую, то их называют дифференциально-алгебраическими уравнениями (ДАУ). Если отсутствует слагаемое с производной, то их принято называть интегро-алгебраическими уравнениями (ИАУ), и такие задачи были впервые исследованы в работах [1, 2]. И тот, и другой класс задач достаточно хорошо изучен. В частности, к настоящему времени уже опубликованы сотни статей и десятки моногра-

фий, посвященных анализу и численному решению ДАУ (см., например, [3] и приводимую там библиографию). Изучение ИАУ началось несколько позже, чем ДАУ. Впервые их качественные свойства и связь с ДАУ были исследованы в работе [2]. Современное состояние данной тематики частично отражено в работах [4–6]. Среди последних публикаций, посвященных ИАУ со слабой особенностью можно отметить [7, 8]. Однако перенос методов и теории, разработанных для ДАУ и ИАУ на вырожденные системы ИДУ не всегда успешен, поэтому их принято выделять в самостоятельный объект исследования, который к настоящему времени изучен недостаточно, что частично подтверждается списком литературы из недавно опубликованной работы [9], содержащей довольно полный обзор по вырожденным системам ИДУ. Основным принципом исследования таких систем является их расщепление на подсистемы ИДУ и уравнений Вольтерра I и II рода с последующим применением кусочно-полиномиальных методов коллокации. Самым малоизученным классом в данном спектре задач остаются вырожденные системы ИДУ типа Вольтерра со слабой особенностью в ядре, которым и посвящена данная работа. К настоящему моменту сделана только одна публикация по этой тематике [10], которая не содержит численного метода решения.

Системы вырожденных ИДУ имеют важное прикладное значение. Они довольно часто возникают при математическом моделировании различных физических и технических процессов. Примеры моделирования различных динамических систем с помощью вырожденных систем ИДУ могут быть найдены в [11], в частности, к ним могут быть сведены математические модели процессов, протекающих в электрических и гидравлических цепях [12–14]. Вырожденные системы ИДУ даже оказываются полезными при моделировании расположения тазобедренного сустава пассажира, сидящего в кресле автомобиля [15] и во многих других случаях, где необходимо моделирование динамики многотельных систем.

Статья организована следующим образом: во введении описаны задачи, приводящие к появлению интегро-дифференциальных уравнений и приводится обзор литературы по данной тематике. В разделе 1 описывается постановка задачи, раздел 2 посвящен условиям существования единственного решения и содержит необходимые дополнительные сведения. В разделе 4 рассматривается численный метод и анализируется его работа на модельных примерах. В заключении содержится краткая сводка результатов, полученных в работе, с указанием направления дальнейших исследований.

1. Постановка задачи

Рассмотрим систему интегро-дифференциальных уравнений

$$A(t)\dot{x}(t) + B(t)x(t) + \int_0^t (t-s)^{-\alpha} K(t,s)x(s)ds = f(t),\ t \in [0,1] = T,\ 0 < \alpha < 1, \qquad (1)$$

с начальными данными

$$x(0) = x_0, \qquad (2)$$

где $A(t), B(t), K(t,s)$ — $(n \times n)$-матрицы, $f(t)$ и $x(t)$ — заданная и искомая n-мерные вектор-функции, x_0 — заданный вектор из $\mathbb{R}^n$. Предполагается, что все входные данные достаточно гладкие в своих областях определения, и, кроме того,

$$\det A(t) = 0\ \forall t \in T. \qquad (3)$$

Такие системы мы называем вырожденными интегро-дифференциальными уравнениями, а под решением начальной задачи (1), (2) будем понимать любую вектор-функцию $x(t)$, которая обращает (1) в тождество и удовлетворяет условию (2). Рассматриваемый класс задач существенно отличается от систем с невырожденной матрицей перед производной искомой вектор-функции. В классической теории, если определитель $\det A(t)$ обращается в ноль в изолированных точках отрезка T, то такие точки называются особыми, т.к. решение в них может не существовать или же они являются точками ветвления решений. Однако, если матрица $A(t)$ вырождена на всем отрезке определения, то в этой области решение может оставаться единственным, а может и не существовать.

2. Существование решения

Приведем ряд известных определений и утверждений.

Определение 1. [22] Пучок постоянных матриц $\lambda A + B,\ \lambda \in \mathbb{C}$ называется регулярным, если существует такое λ, что $\det(\lambda A + B) \neq 0$.

Лемма 1. [22] Пусть пучок постоянных матриц $\lambda A + B$ регулярен. Тогда существуют невырожденные $(n \times n)$-матрицы P и Q с постоянными элементами такие, что

$$P(\lambda A + B)Q = \lambda \begin{pmatrix} E_m & 0 & 0 \\ 0 & E_l & 0 \\ 0 & 0 & N \end{pmatrix} + \begin{pmatrix} J & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & E_k \end{pmatrix}, \tag{4}$$

где $m + l + k = n,\ N, M$ — нильпотентные матрицы размерностей $(k \times k)$ и $(l \times l)$ соответственно, $N^{k_1} = 0,\ M^{l_1} = 0,\ k_1 \leq k,\ l_1 \leq l$.

Определение 2. [19] Выражение $\lambda A(t) + \mu B(t) + C(t),\ t \in T$, где $A(t),\ B(t)$ и $C(t)$ — переменные $(n \times n)$-матрицы, а λ, μ — скалярные параметры, будем называть матричным полиномом.

Определение 3. [19] Говорят, что матричный полином $\lambda A(t) + \mu B(t) + C(t)$ имеет на отрезке T простую структуру, если выполнены следующие условия:

1) все элементы матриц $A(t),\ B(t)$ и $C(t)$ принадлежат C_T^m;
2) $\mathrm{rank} A(t) = k = const\ \forall t \in T$;
3) $\mathrm{rank}(A(t)|B(t)) = k + l = const$;
4) $\det[\lambda A(t) + \mu B(t) + C(t)] = a_0(t)\lambda^k \mu^l + \cdots$, где $a_0(t) \neq 0\ \forall t \in T$.

Лемма 2. [19] Пусть матричный полином $\lambda A(t) + \mu B(t) + C(t)$ имеет на отрезке T простую структуру. Тогда существуют невырожденные для всех $t \in T$ матрицы $R(t)$ и $S(t)$ с элементами из C_T^m такие, что

$$R(t)(\lambda A(t) + \mu B(t) + C(t))S(t) = \\ = \lambda \begin{pmatrix} E_k & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \mu \begin{pmatrix} J_1(t) & 0 & J_2(t) \\ 0 & E_l & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} C_1(t) & C_2(t) & 0 \\ C_3(t) & C_4(t) & 0 \\ 0 & 0 & E_{n-k-l} \end{pmatrix},\ t \in T, \tag{5}$$

где $J_1(t),\ J_2(t),\ C_1(t),\ C_2(t),\ C_3(t),\ C_4(t)$ — некоторые блоки подходящей размерности.

Определение 4. [20, 21] Псевдообратной матрицей к $(m \times n)$-матрице $A(t)$, $t \in T$, называется $(n \times m)$-матрица $A^+(t)$, удовлетворяющая для любых $t \in T$ уравнениям:

$$A(t)A^+(t)A(t) = A(t), \qquad A^+(t)A(t)A^+(t) = A(t),$$

$$(A(t)A^+(t))^\top = A(t)A^+(t), \qquad (A^+(t)A(t))^\top = A^+(t)A(t)$$

Прежде, чем мы сформулируем теорему существования для начальной задачи (1), (2), введем следующие обозначения:

$$\mathcal{B}(t) = (E - A(t)A^+(t))B(t), \qquad \mathcal{K}(t,s) = (E - A(t)A^+(t))K(t,s),$$

$$\zeta(t) = (E - A(t)A^+(t))f(t), \qquad \chi(t) = \frac{d}{dt}\int\limits_0^t (t-s)^{-\alpha}(E - \mathcal{B}(s)\mathcal{B}^+(s))\zeta(s)ds.$$

Теорема 1. [10] Пусть задача (1), (2) удовлетворяет условиям:

1) функции $A(t), B(t), K(t,s)$, $f(t)$ дважды непрерывно-дифференцируемы на отрезке T;
2) $(E - A(0)A^+(0))B(0)x_0 = (E - A(0)A^+(0))\zeta(0)$;
3) $\pi \sin \alpha\pi (E - \mathcal{B}(0)\mathcal{B}^+(0))\mathcal{K}(0,0)x_0 = (E - \mathcal{B}(0)\mathcal{B}^+(0))\chi(0)$;
4) матричный полином $\lambda A(t) + \mu B(t) + K(t,t)$ имеет на отрезке T простую структуру.

Тогда существует единственное непрерывно-дифференцируемое решение задачи (1), (2) на T.

Отметим, что третье условие теоремы обеспечивает совместность начальных данных (2) с правой частью системы (1). Таким образом, теоретически, теорема 1 обеспечивает возможность расщепления системы (1) с помощью серии невырожденных преобразований на три невырожденных подсистемы: систему ИДУ в нормальной форме, систему уравнений Вольтерра первого рода и системы уравнений Вольтерра первого рода. Практически это возможно сделать далеко не всегда. Рассмотрим случай, когда $A(t) \equiv 0$, и B, K — постоянные матрицы, т.е. система (1) имеет вид:

$$Bx(t) + \int\limits_0^t (t-s)^{-\alpha} Kx(s)ds = f(t). \tag{6}$$

Если пучок $\lambda B + K$ регулярен (см. определение 1), то решение системы (6) может быть получено в явной форме. Для этого умножим систему (6) справа на матрицу P и введем замену $x = Qy$, где P и Q — матрицы из леммы 1. Получим

$$\begin{pmatrix} E_m & 0 & 0 \\ 0 & E_l & 0 \\ 0 & 0 & N \end{pmatrix} \begin{pmatrix} y_1(s) \\ y_2(s) \\ y_3(s) \end{pmatrix} + \int\limits_0^t \begin{pmatrix} J & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & E_k \end{pmatrix} (t-s)^{-\alpha} \begin{pmatrix} y_1(s) \\ y_2(s) \\ y_3(s) \end{pmatrix} ds = \begin{pmatrix} \phi_1(t) \\ \phi_2(t) \\ \phi_3(t) \end{pmatrix},$$

где $(y_1^\top(t), y_2^\top(t), y_3^\top(t))^\top = y(t)$, $(\phi_1^\top(t), \phi_2^\top(t), \phi_3^\top(t))^\top = Pf(t)$. Таким образом, исходная система разбилась на 3 подсистемы, для каждой из которых решение можно выписать в явном виде. Первая подсистема представляет собой систему интегральных уравнений Абеля второго рода и ее решение может быть найдено в терминах функции Миттаг-Леффлера [23]:

$$y_1(t) = \frac{d}{dt}\int\limits_0^t \mathcal{M}_{1-\alpha}[J\Gamma(\alpha)(t-s)^{1-\alpha}]\phi_1(s)ds. \tag{7}$$

Решение второй подсистемы легко найти, последовательно исключая неизвестные:

$$y_2(t) = \phi_2 - UM\phi_2 + U^2M^2\phi_2 + \cdots + (-1)^{l-1}U^{l_1-1}M^{l_1-1}\phi_2, \tag{8}$$

$$U\psi = \int_0^t (t-s)^{-\alpha}\psi(s)ds.$$

Решение третьей подсистемы определяется структурой решения уравнения Абеля первого рода [23]:

$$y_3(t) = WN^0\phi_3 - W^2N^1\phi_3 + \cdots + (-1)^kW^{k_1}N^{k_1-1}\phi_3, \tag{9}$$

$$W\psi = \frac{sin\alpha\pi}{\pi}\frac{d}{dt}\int_0^t \frac{\psi(s)}{(t-s)^{1-\alpha}}ds.$$

Таким образом решение $x(t)$ может быть найдено как $x(t) = Qy(t)$, где $y(t)$ определяется по формулам (7), (8), (9).

Для иллюстрации приведем пример.

Пример 1.

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} x(t) + \int_0^t (t-s)^{-\alpha} \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} x(s) = f(t). \tag{10}$$

Заменой $x = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{-1} y$ задача (10) сводится к системе

$$Ny + \int_0^t (t-s)^{-\alpha}y(s)ds = f(t),\ N^3 = 0,$$

решение которой находится последовательным применением оператора (7).

3. Численный метод

Для численного решения начальной задачи (1), (2) была предложена и реализована следующая разностная схема первого порядка:

$$A_{i+1}(x_{i+1} - x_i) + hB_{i+1}x_{i+1} + h^2\sum_{j=1}^{i+1}\omega_{i+1,j}K_{i+1,j}x_j = hf_{i+1},\ i = 0, 1, 2, \ldots, \mathcal{N}, \tag{11}$$

где $\mathcal{N}$ — число узлов сетки, $h = 1/N$,

$$A_{i+1} = A(t_{i+1}),\ B_{i+1} = B(t_{i+1}),\ K_{i+1,j} = K(t_{i+1}, t_j),\ t_i = ih,$$

а веса $\omega_{i+1,j}$ находятся по формуле

$$\omega_{i+1,j} = \int_{(j-1)h}^{jh} (t_{i+1} - s)^{-\alpha}ds = \frac{h^{1-\alpha}}{1-\alpha}\left[(i-j+2)^{1-\alpha} - (i-j+1)^{1-\alpha}\right].$$

Теорема 2. Пусть выполнены условия теоремы 1. Тогда

1) начиная с некоторого $h \le h_*$ система (11) имеет решение $\forall i \in \{0, \mathcal{N}\}$;
2) справедлива оценка

$$\max_{i=\overline{0,N}} |x_i - x(t_i)| \le \kappa h^{\min(\alpha, 1-\alpha)}, \ \kappa = const > 0.$$

Доказательство. Перепишем равенство (11) в виде

$$\left(A_{i+1} + hB_{i+1} + h^2\omega_{i+1,i+1}K_{i+1,i+1}\right) x_{i+1} = A_i x_i + h^2 \sum_{j=1}^{i} \omega_{i+1,j} K_{i+1,j} x_j = h f_{i+1}. \qquad (12)$$

Согласно теореме 1, начиная с некоторого $h \le h_*$, матричный полином $A_{i+1} + hB_{i+1} + h^2\omega_{i+1,i+1}K_{i+1,i+1}$ имеет простую структуру и не обращается в нуль. Далее, перепишем систему (11) в виде

$$A_{i+1}\frac{x_{i+1} - x_i}{h} + B_{i+1}x_{i+1} + h\sum_{j=1}^{i+1} \omega_{i+1,j} K_{i+1,j} x_j = f_{i+1},$$

введем замену переменной $y_{i+1} = S_{i+1}x_{i+1}$ и умножим полученную систему на R_{i+1}, где $S_i = S(t_i)$, $R_i = R(t_i)$, а $R(t)$ и $S(t)$ — матрицы из леммы 2. Таким образом, мы расщепили (12) на три подсистемы разностных уравнений, соответствующих системе ИДУ в нормальной форме, системе уравнений Вольтерра второго рода и системе уравнений Вольтерра первого рода, для каждой из которых сходимость доказана в классической теории (см., например, [6]).

□

Численный метод (11) был реализован в пакете прикладных программ MATLAB. Для анализа работы метода были использованы специально построенные модельные примеры.

Пример 2.

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \dot{x}(t) + \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} x(t) + \int_0^t (t-s)^{-1/3} \begin{pmatrix} t & 0 & 0 \\ 0 & t & 0 \\ 0 & 0 & 1 \end{pmatrix} x(s)ds = \begin{pmatrix} 1 + \frac{3}{2}t^{5/3} \\ t + \frac{9}{10}t^{8/3} \\ \frac{27}{40}t^{8/3} \end{pmatrix},$$

$$x(0) = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \ t \in [0, 1].$$

Легко проверить, что данная начальная задача удовлетворяет теореме 1. Здесь точное решение известно и имеет вид $x(t) = (1, t, t^2)^\top$. Результаты расчета приведены в таблице:

Обозначения: Nset — число узлов сетки, err_j — максимальная погрешность по каждой компоненте $x_j(t)$, $j = 1, 2, 3$.

Таблица

Результаты расчета для примера 2

Nset	err_1	err_2	err_3
90	0.010	0.011	0.023
270	0.005	0.008	0.016
810	0.003	0.004	0.009
2430	0.001	0.002	0.005

Если условия теоремы 1 не выполнены, то метод (11) неустойчив либо принципиально не применим. Чтобы проиллюстрировать сложность и неоднозначность изучаемых задач, рассмотрим следующий пример.

Пример 3. Для нижеприведенной системы не выполняется условие простой структуры:

$$\begin{pmatrix} 1 & t & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \dot{x}(t) + \begin{pmatrix} 0 & d & 0 \\ 1 & t & 0 \\ 0 & 0 & 0 \end{pmatrix} x(t) + \int_0^t (t-s)^{-\alpha} \begin{pmatrix} 0 & 0 & t \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} x(s)ds = f(t), t \in [0,1], \quad (13)$$

$$f(t) = (f_1(t), f_2(t), f_3(t))^\top,\ x(t) = (x_1(t), x_2(t), x_3(t))^\top.$$

При исследовании разрешимости системы (13) возникают следующие ситуации:

1) $d \neq 1$: решение существует и единственно $\forall f(t) \in C^2_{[0,1]}$;
2) $d = 1$: система (13) разрешима тогда и только тогда, когда $f_1(t) - tf_3(t) - \dot{f}_2(t) = 0$. При этом одну из компонент ($x_1(t)$ или $x_2(t)$) мы можем взять в виде произвольной вектор-функции из $C^1_{[0,1]}$.

В то же время, при применении для решения системы (13) разностной схемы (11) необходимо учесть следующее:

1) $|d| < 1$: метод (11) неустойчив, т.к. справедлива оценка $x_{i+1} = \mathcal{O}\left(\frac{1}{d^i}\right)$;
2) $d = 0$: решение системы (13) существует, но $\det\left(A_{i+1} + hB_{i+1} + h^2\omega_{i+1,i+1}K_{i+1,i+1}\right) = 0\ \forall h$;
3) $d = 1$: $\det\left(A_{i+1} + hB_{i+1} + h^2\omega_{i+1,i+1}K_{i+1,i+1}\right) \neq 0\ \forall h$, но система (13) может не иметь решения.

Заключение

В статье на основе теории матричных пучков и фактов из теории численного решения интегральных уравнений проанализированы условия существования и единственности решения вырожденных систем ИДУ со слабой особенностью в ядре. К подобным математическим формулировкам приводит моделирование процессов, протекающих в электрических и гидравлических цепях, различных динамических системах, в частности, многотельных. Поэтому исследование и численное решение такого рода задач является актуальным, а результаты исследований — востребованными на практике. В работе было показано, что в случае постоянных матриц решение вырожденной системы ИДУ можно выписать в явном виде в терминах функции Миттаг-Леффлера. Предложен численный метод решения таких уравнений, который был реализован в пакете прикладных программ MATLAB и протестирован на модельных примерах. Показано, что в случае нарушения условий теоремы

существования данный метод неприменим. Следующим направлением исследований является качественный анализ ИДУ со слабой особенностью в ядре, в которое искомая функция входит нелинейно, а также изучение работоспособности численного метода на реальных задачах из приложений.

Работа выполнена при поддержке РФФИ, проект № 20-51-54003.

Литература

1. Gear C.W. Differential algebraic equations, indices, and integral algebraic equations // SIAM Journal of Numerical Analysis. 1990. Vol. 27, no. 6. P. 1527–1534. DOI: 10.1137/0727089.
2. Чистяков В.Ф. О сингулярных системах обыкновенных дифференциальных уравнений и их интегральных аналогах // Функции Ляпунова и их применения. Новосибирск: Наука, 1987. С. 231–239.
3. Lamour R., Marz R., Tischendorf C. Differential-algebraic equations: a projector based analysis. Berlin, Heidelberg: Springer-Verlag, 2013. 649 p. DOI: 10.1007/978-3-642-27555-5.
4. Brunner H., van der Houwen P.J. The numerical solution of Volterra equations (CWI Monographs 3). Elsevier Science Ltd, 1986. 604 p.
5. Brunner H. Collocation methods for Volterra integral and related functional equations. Cambridge University Press, 2004. 612 p. DOI: 10.1017/CBO9780511543234.
6. Brunner H. Volterra Integral Equations: An Introduction to Theory and Applications. Cambridge University Press, 2017. 402 p. DOI: 10.1017/9781316162491.
7. Liang H., Brunner H. On the convergence of collocation solutions in continuous piecewise polynomial spaces for weakly singular Volterra integral equations // SIAM Journal on Numerical Analysis. 2019. Vol. 57, no. 4. P. 1875–1896. DOI: 10.1007/s10543-016-0609-x.
8. Sajjadi S.A., Pishbin S. Convergence analysis of the product integration method for solving the fourth kind integral equations with weakly singular kernels // Numerical Algorithms. 2021. No. 86. P. 25–54. DOI: 10.1007/s11075-020-00877-x.
9. Liang H., Brunner H. Collocation methods for integro-differential algebraic equations with index 1 // IMA Journal of Numerical Analysis. 2020. No. 39. P. 850–885. DOI: 10.1093/imanum/drz01.
10. Bulatov M.V., Lima P.M., Weinmuller E.B. Existence and uniqueness of solutions to weakly singular integral-algebraic and integro-differential equations // Central European Journal of Mathematics. 2014. Vol. 12. P. 308–321. DOI: 10.2478/s11533-013-0334-5.
11. Doležal V. Dynamics of Linear Systems. Prague: Academia Publishing House of the Czechoslovak Academy of Sciences, 1967. 325 p.
12. Jiang Y.L., Wing O. Waveform relaxation of linear integral-differential equations of circuit simulation // IEEE Design Automation Conference. 1999. P. 61–64. DOI: 10.1109/aspdac.1999.759710.
13. Ушаков Е.И. Статическая устойчивость электрических систем. Новосибирск: Наука, 1988. 273 с.
14. Nassirharand A. A new technique for solving sets of coupled nonlinear algebraic and integro-differential equations encountered in hydraulics // International Journal of Contemporary Mathematical Sciences. 2008. Vol. 3, no. 33. P. 1611–1617.

15. Ippili R.K., Davies P., Bajaj A.K., Hagenmeyer L. Nonlinear multi-body dynamic modeling of seat–occupant system with polyurethane seat and H-point prediction // International Journal of Industrial Ergonomics. 2008. Vol. 38, no. 5. P. 368–383. DOI: 10.1016/j.ergon.2007.08.014.

16. Чистякова Е.В. О свойствах разностных схем для вырожденных интегродифференциальных уравнений индекса 1 // Журн. вычисл. матем. и матем. физ. 2009. Т. 49, № 9. С. 1579–1588.

17. Булатов М.В., Чистякова Е.В. Об одном семействе вырожденных интегродифференциальных уравнений // Журн. вычисл. математики и мат. физики. 2011. Т. 51, № 9. С. 1665–1673.

18. Банг Н.Д., Чистяков В.Ф., Чистякова Е.В. О некоторых свойствах вырожденных систем линейных интегро-дифференциальных уравнений. I // Известия Иркутского государственного университета. Серия «Математика». 2015. № 11. С. 13–27.

19. Булатов М.В., Ли М.-Г. Применение матричных полиномов к исследованию линейных дифференциально-алгебраических уравнений высокого порядка // Дифференциальные уравнения. 2008. Т. 44, № 10. С. 1299–1306.

20. Бояринцев Ю.Е. Регулярные и сингулярные системы линейных обыкновенных дифференциальных уравнений. Новосибирск: Наука, 1980. 222 с.

21. Lancaster P. Theory of Matrices. Academic Press, 1985. 570 p.

22. Гантмахер Ф.Р. Теория матриц. М.: Наука, 1967. 577 с.

23. Краснов М.Л. Интегральные уравнения: введение в теорию. М.: Наука, 1975. 302 с.

Чистякова Елена Викторовна, к.ф.-м.н., н.с., Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)

Соловарова Любовь Степановна, к.ф.-м.н., м.н.с., Институт динамики систем и теории управления им. В.М. Матросова СО РАН (Иркутск, Российская Федерация)

Доан Тай Сон, д.ф.-м.н., доцент, Институт математики Вьетнамской академии наук и технологий (Ханой, Вьетнам)

DOI: 10.14529/cmse210301

A NUMERICAL METHOD FOR SOLVING SINGULAR INTEGRAL ALGEBRAIC EQUATIONS WITH WEAKLY SINGULAR KERNELS

© 2021 E.V. Chistyakova[1], L.S. Solovarova[1], Doan Thai Son[2]

[1]*Institute for System Dynamics and Control Theory SB RAS*
(str. Lermontova 134, Irkutsk, 664033 Russia),
[2]*Institute of Mathematics of the Vietnam Academy of Science and Technology*
(str. Hoang Quoc Viet 18, Hanoi, 10307 Vietnam)
E-mail: elena.chistyakova@icc.ru, soleilu@mail.ru, dtson@math.ac.vn

Received: 02.03.2021

Statements of many applied problems often include differential equations and Volterra integral equations of the first and second kind. By joining such equations together, we obtain a system of integral differential equations with a singular matrix multiplying the leading part. Such systems are commonly referred to as singular integral differential equations. If they do not contain an integral part, then they are called differential-algebraic equations. If there is no term with a derivative, then they are usually called integral algebraic equations. Such mathematical problem statements arise in simulation of processes occurring in electrical and hydraulic circuits, various dynamic systems, in particular, multibody systems. Therefore, qualitative study and numerical solution of such problems are quite relevant, and the results of research remain in demand in practice. In this paper, on the basis of the theory of matrix pencils, as well as using research schemes developed for differential algebraic and integral algebraic equations, the conditions for the existence and uniqueness of the solution of singular integral-differential equations with a weakly singular kernels are analyzed and a numerical method for their solution is proposed. The method was coded in MATLAB and tested on model examples.

Keywords: differential equations, integral differential equations, Abel equation, weak singularity.

FOR CITATION

Chistyakova E.V., Solovarova L.S., Doan Thai Son. A Numerical Method for Solving Singular Integral Algebraic Equations with Weakly Singular Kernels. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2021. Vol. 10, no. 3. P. 5–15. (in Russian) DOI: 10.14529/cmse210301.

References

1. Gear C.W. Differential algebraic equations, indices, and integral algebraic equations. SIAM Journal of Numerical Analysis. 1990. Vol. 27, no. 6. P. 1527–1534. DOI: 10.1137/0727089.

2. Chistyakov V.F. On singular systems of ordinary differential equations and their integral analogs. Lyapunov functions and their applications. Novosibirsk, Nauka Publishing House, 1987. P. 231–239. (in Russian)

3. Lamour R., Marz R., Tischendorf C. Differential-algebraic equations: a projector based analysis. Berlin, Heidelberg, Springer-Verlag, 2013. 649 p. DOI: 10.1007/978-3-642-27555-5.

4. Brunner H., van der Houwen P.J. The numerical solution of Volterra equations (CWI Monographs 3). Elsevier Science Ltd, 1986. 604 p.

5. Brunner H. Collocation methods for Volterra integral and related functional equations. Cambridge University Press, 2004. 612 p. DOI: 10.1017/CBO9780511543234.

6. Brunner H. Volterra Integral Equations: An Introduction to Theory and Applications. Cambridge University Press, 2017. 402 p. DOI: 10.1017/9781316162491.

7. Liang H., Brunner H. On the convergence of collocation solutions in continuous piecewise polynomial spaces for weakly singular Volterra integral equations. SIAM Journal on Numerical Analysis. 2019. Vol. 57, no. 4. P. 1875–1896. DOI: 10.1007/s10543-016-0609-x.

8. Sajjadi S.A., Pishbin S. Convergence analysis of the product integration method for solving the fourth kind integral equations with weakly singular kernels. Numerical Algorithms. 2021. No. 86. P. 25–54. DOI: 10.1007/s11075-020-00877-x.

9. Liang H., Brunner H. Collocation methods for integro-differential algebraic equations with index 1. IMA Journal of Numerical Analysis. 2020. No. 39. P. 850–885. DOI: 10.1093/imanum/drz01.

10. Bulatov M.V., Lima P.M., Weinmuller E.B. Existence and uniqueness of solutions to weakly singular integral-algebraic and integro-differential equations. Central European Journal of Mathematics. 2014. Vol. 12. P. 308–321. DOI: 10.2478/s11533-013-0334-5.

11. Doležal V. Dynamics of Linear Systems. Prague, Academia Publishing House of the Czechoslovak Academy of Sciences, 1967. 325 p.

12. Jiang Y.L., Wing O. Waveform relaxation of linear integral-differential equations of circuit simulation. IEEE Design Automation Conference. 1999. P. 61–64. DOI: 10.1109/aspdac.1999.759710.

13. Ushakov Ye.I. Static Stability of Electrical Systems. Novosibirsk, 1988. 273 p.

14. Nassirharand A. A new technique for solving sets of coupled nonlinear algebraic and integro-differential equations encountered in hydraulics. International Journal of Contemporary Mathematical Sciences. 2008. Vol. 3, no. 33. P. 1611–1617.

15. Ippili R.K., Davies P., Bajaj A.K., Hagenmeyer L. Nonlinear multi-body dynamic modeling of seat–occupant system with polyurethane seat and H-point prediction. International Journal of Industrial Ergonomics. 2008. Vol. 38, no. 5. P. 368–383. DOI: 10.1016/j.ergon.2007.08.014.

16. Chistyakova E.V. Properties of finite-difference schemes for singular integrodifferential equations of index 1. Comput. Math. Math. Phys. 2009. Vol. 49, no. 9. P. 1507–1515 DOI: 10.1134/S096554250909005X.

17. Bulatov M.V., Chistyakova E.V. On a family of singular integro-differential equations. Comput. Math. Math. Phys. 2011. Vol. 51, no. 9. P. 1558–1566. DOI: 10.1134/S0965542511090065.

18. Bang N.D., Chistyakov V.F., Chistyakova E.V. On some properties of degenerate systems of linear integro-differential equations. I. Bulletin of the Irkutsk State University. Series "Mathematics". 2015. No. 11. P. 13–27. (in Russian)

19. Bulatov M.V., Ming-Gong Lee. Application of matrix polynomials to the analysis of linear differential-algebraic equations of higher order. 2008. Vol. 44, no. 10. P. 1299–1305.

20. Boyarintsev Yu.E. Regular and singular systems of linear ordinary differential equations. Novosibirsk, Nauka Publishing House, 1980. 222 p. (in Russian)

21. Lancaster P. Theory of Matrices. Academic Press, 1985. 570 p.

22. Gantmacher F.R. Matrix theory. Moscow, Nauka Publishing House, 1967. 577 p. (in Russian)

23. Krasnov M.L. Integral equations: an introduction to the theory. Moscow, Nauka Publishing House, 1975. 302 p. (in Russian)

УДК 004.89, 004.048 **DOI: 10.14529/cmse210302**

ОЧИСТКА СЕНСОРНЫХ ДАННЫХ В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ УПРАВЛЕНИЯ ОТОПЛЕНИЕМ ЗДАНИЙ

© 2021 М.Л. Цымблер, Я.А. Краева, Е.А. Латыпова, Е.В. Иванова, Д.А. Шнайдер, А.А. Басалаев

Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76)
E-mail: mzym@susu.ru, kraevaya@susu.ru, latypovaea@susu.ru, elena.ivanova@susu.ru, shnaiderda@susu.ru, basalaevaa@susu.ru

Поступила в редакцию: 03.09.2020

В современных интеллектуальных системах управления отоплением зданий зачастую возникают пропуски значений или выбросы в показаниях температурных и других датчиков ввиду сбоев программного или аппаратного обеспечения либо человеческого фактора. Для обеспечения эффективного анализа данных и принятия решений некорректные данные датчиков следует очищать путем восстановления пропущенных значений и сглаживания выбросов. В данной статье представлен пример SCADA-системы ПолиТЭР для управления отоплением, установленной в Южно-Уральском государственном университете, и описана структура и принципы реализации Модуля очистки данных, внедренного в указанную систему. Модуль очистки данных реализован с помощью технологий интеллектуального анализа данных и нейронных сетей в виде набора следующих подсистем. Препроцессор извлекает необработанные данные из хранилища данных системы и подготавливает обучающий набор данных для дальнейшей обработки. Предиктор представляет собой рекуррентную нейронную сеть для прогнозирования следующего значения датчика на основе его исторических данных. Реконструктор определяет, является ли текущее значение датчика выбросом, и в таком случае заменяет его на синтетическое значение, полученное Предиктором. Наконец, Детектор аномалий в режиме реального времени обнаруживает аномальные промежутки в данных датчика. В вычислительных экспериментах на реальных данных разработанный модуль показал относительно высокую и стабильную точность, а также адекватное обнаружение аномалий.

Ключевые слова: умный дом, температурный датчик, управление отоплением, хранилище данных, очистка данных, временной ряд, восстановление пропущенных значений, поиск выбросов, обнаружение аномалий, рекуррентная нейронная сеть.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Цымблер М.Л., Краева Я.А., Латыпова Е.А., Иванова Е.В., Шнайдер Д.А. Басалаев А.А. Очистка сенсорных данных в интеллектуальных системах управления отоплением зданий // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 3. С. 16–36. DOI: 10.14529/cmse210302.

Введение

Современные интеллектуальные системы управления отоплением зданий используют различные источники данных, включая данные измерений счетчиков потребления коммунальных услуг, данные от контроллеров процесса, датчиков внутреннего климата и др. Внедрение технологий Интернета вещей (IoT, Internet of Things) в системы такого типа дополнительно позволяет получать большие массивы данных о различных параметрах, влияющих на общие условия отопления в здании. Полученный в результате массив больших данных позволяет проводить всесторонний анализ систем отопления, позволяя своевременно выявлять случаи отклоняющихся значений производительности и показателей энергоэффективности зданий за допустимые пределы. Однако, в соответствии с теорией

надежности при рассмотрении всей совокупности данных в единой системе увеличение количества источников данных может привести к выполнению некорректного анализа данных в случае сбоя даже одного источника данных.

Причиной возникновения некорректных показаний IoT-датчиков могут являться различные факторы: износ оборудования, неправильные монтаж и эксплуатация оборудования (человеческий фактор), конструктивные недостатки программного и аппаратного обеспечения, выход условий эксплуатации за пределы допустимых значений и др. Некорректные показания IoT-датчиков являются причиной снижения энергоэффективности эксплуатации объектов и их некорректного управления в автоматическом и автоматизированном режиме, а также приводят к штрафным санкциям от поставщиков энергоносителей. В соответствии с этим актуальной задачей является разработка методов и алгоритмов очистки данных, обеспечивающих оперативное выявление некорректных показаний датчиков и восстановление утерянных данных.

В настоящей статье рассматривается проблема очистки данных в интеллектуальной системе управления теплоснабжением зданий кампуса Южно-Уральского государственного университета (ЮУрГУ). В кампусе ЮУрГУ реализована интеллектуальная система управления теплоснабжением зданий на основе SCADA-системы ПолиТЭР [1], которая позволяет выполнять мониторинг инженерных систем зданий кампуса и управление режимами их работы, включая проводные и беспроводные IoT-датчики. Авторами статьи разработан и внедрен в систему ПолиТЭР Модуль очистки данных, обеспечивающий в режиме реального времени выявление некорректных показаний и восстановление утерянных данных IoT-датчиков.

Статья организована следующим образом. В разделе 1 приведен обзор SCADA-системы «ПолиТЭР». В разделе 2 приведен краткий обзор научных работ в данной области. Раздел 3 посвящен описанию архитектуры и принципов реализации Модуля очистки данных для SCADA-системы «ПолиТЭР». Результаты вычислительных экспериментов, исследующих эффективность разработанного модуля, представлены в разделе 4. В заключении подводятся итоги проведенного исследования.

1. Обзор системы ПолиТЭР

На рис. 1 представлена структура интеллектуальной системы управления теплоснабжением зданий кампуса ЮУрГУ. На нижнем уровне система включает в себя различные проводные и беспроводные датчики, приборы учета и контроллеры. На среднем уровне выполняется обеспечение связи контроллеров и узлов учета с сервером баз данных посредством различного проводного и беспроводного сетевого оборудования. Третий уровень включает SCADA-систему «ПолиТЭР» с сервером баз данных, осуществляющую обработку информации. Представление обработанных данных осуществляется на АРМ локальных пользователей через ЛВС ЮУрГУ или на АРМ удаленных пользователей через сеть Интернет.

Внедрение данной системы в ЮУрГУ началось в 2010 г. Особое внимание при внедрении данной системы в ЮУрГУ было уделено системе теплоснабжения. Целью внедрения стала комплексная оптимизация процессов управления теплоснабжением и теплопотреблением за счет интеграции территориально распределенного измерительного оборудования с автоматическими системами управления собственными системами когенерации и распределительными системами как централизованно, так и индивидуально на отдельных потребителях.

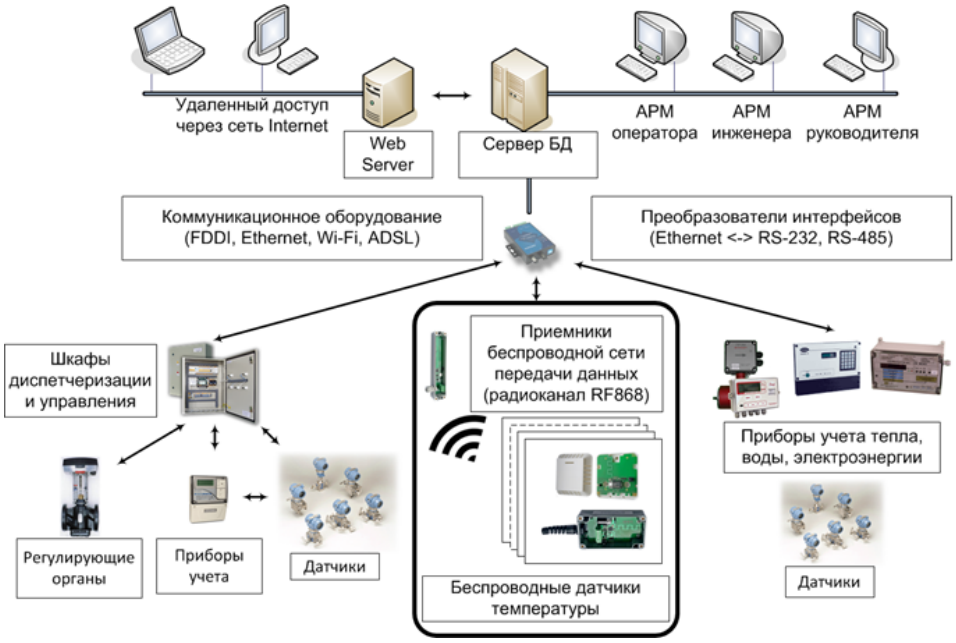


Рис. 1. Архитектура SCADA-системы ПолиТЭР

Интеллектуальной системы управления теплоснабжением зданий включают в себя следующее измерительное и управляющее оборудование:

- узлы учета тепловой энергии (57 шт.), оборудованные тепловычислителем с подключенными к нему 2 датчиками давления, 2 датчиками температуры и 2 расходомерами;
- узлы учета холодной воды (58 шт.), оборудованные вычислителем с подключенными к нему датчиком давления и расходомером;
- узлы учета газа (3 шт.), оборудованные газовым корректором с подключенными к нему расходомером датчиками температуры и давления;
- контроллеры управления распределением тепловой энергии в индивидуальных тепловых пунктах (24 шт.) с подключенными к ним 5 датчиками температуры и различными дискретными датчиками защит, а также управляющими приводами.
- контроллеры управления генерацией и распределением тепловой и электрической энергии (5 шт.) когенерационных установок, газовой котельной и центральных тепловых пунктов с множеством различных датчиков и управляющих устройств.
- беспроводные датчики контроля температурного режима помещений (более 300 шт.).

Программный комплекс ПолиТЭР реализован на языках программирования C++, R и использует Oracle в качестве базовой СУБД. Поддерживаются открытые и проприетарные протоколы обмена данными с оборудованием различных производителей, конфигурируемая среда визуализации данных, оповещения оператора системы по SMS и электронной почте.

В 2018 г. сенсорная подсистема была существенно расширена IoT-устройствами (установлено более 300 беспроводных датчиков температуры), позволяющими собирать дополнительную информацию о температурном режиме помещений, что позволило существенно оптимизировать систему отопления зданий. Внедрение интеллектуальной системы управления теплоснабжением зданий в кампусе ЮУрГУ позволило в 2018 г. сэкономить около 15 % тепловой энергии.

Основной проблемой эксплуатации измерительного оборудования является периодическое появление ошибочных показаний или данных с наличием возмущений неизвестного характера. Одной из причин является выход оборудования из строя: ввиду устаревания или нарушения условий эксплуатации оборудования ежегодно 10–20 различных устройств подлежат замене. Другой причиной является засорение областей контакта датчика с измерительной средой. Также периодически возникают обрывы линий связи с контрольно-измерительными устройствами, при этом большинство приборов, в особенности, контроллеры, не поддерживают аппаратные архивы данных. Типичной причиной также является некорректная установка оборудования, приводящая к возникновению дополнительных возмущений, связанных с нетиповыми режимами эксплуатации IoT-датчиков в области их установки.

Некорректные данные приводят к следующим основным проблемам:

– ошибочные вычисления в ходе выполнения оптимизационных алгоритмов, построению некорректных характеристик объектов управления и, как следствие, к некорректному принятию стратегических решений;
– некорректное принятие организационно-управляющих решений персоналом при наличии недостоверных отклонений параметров эксплуатации от их номинальных значений;
– некорректное автоматическое управление посредством контроллеров может привести не только к перерасходу энергии, но и к выходу из строя инженерных коммуникаций (например, замораживание системы отопления);
– некорректный расчет потребляемых энергоресурсов или наложение энергоснабжающими организациями на потребителя штрафных санкций за несвоевременное обнаружение и устранение неисправностей.

Своевременное обнаружение, устранение неисправностей оборудования и восстановление данных дает существенный организационно-экономический эффект за счет уменьшения или исключения упущенной выгоды при возникновении указанных проблем.

В описываемом исследовании предложен Модуль очистки данных, расширяющий систему ПолиТЭР и решающий следующие основные задачи:

– оперативное обнаружение аномального поведения датчиков температуры и уведомление оператора о найденных аномалиях;
– оперативное обнаружение пропусков и выбросов в измеряемых данных датчиков и замена таких значений на синтетические правдоподобные данные.

Разработанный Модуль очистки данных располагается в системе ПолиТЭР между уровнем считывания данных и уровнем их использования для аналитических расчетов и представления. При этом отображаемые данные отмечаются как исходные или восстановленные, а доступ к некорректным данным сохраняется для возможности их более глубокого анализа.

2. Обзор работ

Обнаружению выбросов посвящено множество различных методик, многие из которых используются в теории обнаружения отказов оборудования [25, 28]. К базовым подходам относятся методы, определяющие выход за границы допустимого диапазона отклонения фактических значений от значений, рассчитанных по модели объекта [24, 30].

В работе [26] на примере систем мониторинга вентиляции и кондиционирования для определения некорректных показаний используется сочетание трех техник обработки данных: уменьшение размерности коррелирующих данных, сигнализирующих о возникновении выбросов (метод ReliefF и адаптивные генетические алгоритмы); применение расширенного фильтра Калмана для фильтрации шумов и разложения данных во временные ряды; разделение областей корректных и некорректных значений статических и динамических параметров модели объекта с использованием рекурсивного одноклассового метода опорных векторов.

В работе [2] рассматривается задача прогнозного управления теплоснабжением потребителей с помощью нейросети, обученной на статистических данных, получаемых с подсистемы мониторинга. Искусственная нейронная сеть рассматривается в качестве основного инструмента, минимизирующего ошибки, связанные с ручным управлением температурой на выходе из котельной.

Исследование подходов к сбору и анализу данных на предприятиях водоснабжения и водоотведения рассмотрены в работе [3].

Для определения отклонений в работе систем централизованного теплоснабжения в работе [13] предлагается применять метод ближайшего соседа. В данной работе в качестве эталона берутся параметры работы группы схожих автоматизированных индивидуальных тепловых пунктов (АИТП). Неисправностями считаются отклонения показателей работы одного АИТП от показателей группы схожих с ним АИТП, подключенных к одной системе централизованного теплоснабжения, на величину заданного порога.

В работе [15] представлено применения метода главных компонент для идентификации выбросов в данных работы систем отопления, вентиляции и кондиционирования. В работе [12] описан метод восстановления данных, применяемый совместно с методом главных компонент.

Также стоит отметить метод построения балансовых моделей, применяемый к связанным теплоэнергетическим комплексам. Расхождение энергетического или гидравлического баланса в таких моделях свидетельствует о наличии неисправностей в системе или о наличии выбросов [17].

Восстановление данных, утраченных при возникновении выбросов, выполняется путем их расчета на базе методов прогнозирования. Множество работ посвящено прогнозированию параметров систем теплоснабжения с использованием статических и динамических факторных моделей и моделей временных рядов с идентификацией методами авторегрессионного анализа [29], метода опорных векторов [16] и рекуррентных нейронных сетей [5].

Также интерес представляет подход к восстановлению статических и динамических характеристик моделей, искажение которых возникает в результате периодических возмущений. Учет подобных возмущений выполняется путем описания их времени действия индикаторными функциями [8].

При этом необходимо отметить, что особенностью эксплуатационных данных систем теплоснабжения является периодический характер, связанный с цикличностью изменения погодных условий и режимов эксплуатации помещений в зависимости от времени суток. Выбросы в эксплуатационных данных, возникающие при работе различных подсистем, связанных с теплоснабжением, характеризуются резким изменением измеряемого сигнала, длящимся в пределах ограниченного периода времени.

Ввиду этого в данной статье предлагается использовать рекуррентные сети долгой краткосрочной памяти как для определения последовательностей подобных выбросов, так и для восстановления утраченных последовательностей эксплуатационных измерительных данных, носящих циклический характер изменения.

3. Разработка Модуля очистки данных

3.1. Общая архитектура

Общая архитектура Модуля очистки данных представлена на рис. 2. Модуль тиражируется для каждого температурного датчика системы ПолиТЭР и состоит из следующих основных подсистем: Препроцессор, Предиктор, Реконструктор и Детектор аномалий. Препроцессор подготавливает накопленные показания датчиков для дальнейшей обработки. Предиктор обеспечивает искусственную нейронную сеть, которая выполняет прогноз следующего значения датчика на основе его исторических данных. Реконструктор определяет, является ли текущее показание датчика выбросом, и в этом случае заменяет его на синтетическое значение, полученное Предиктором. Детектор аномалий обнаруживает аномальные подпоследовательности в показаниях датчика для последующего уведомления оператора системы ПолиТЭР.

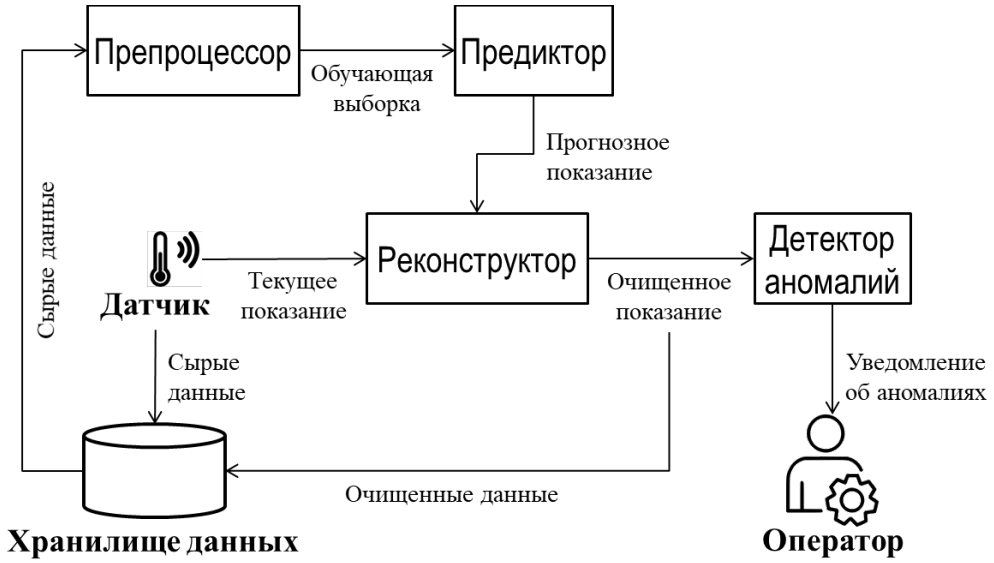


Рис. 2. Архитектура Модуля очистки данных

Технологический цикл Модуля очистки данных для некоторого датчика выглядит следующим образом. Препроцессор выполняет свои действия регулярно с частотой, определяемой оператором системы (типичным случаем является запуск Препроцессора один раз в месяц). Препроцессор извлекает из хранилища данных неочищенные показания датчика, накопленные к текущему моменту, и подготавливает обучающую выборку для нейронной сети подсистемы Предиктор.

Предиктор также запускается регулярно, но в соответствии с частотой снятия показаний датчика (например, один раз в 10 минут), и выполняет следующие действия. Нейронная сеть прогнозирует текущее значение датчика. Если при этом датчик возвратил пустое значение, то оно заменяется на прогнозное. Иначе подсистема Реконструктор выполняет бинарную классификацию значения, полученного от датчика: выброс или норма.

Если текущее значение распознается как выброс, Реконструктор заменяет текущее значение на прогнозное. Модуль очистки данных передает прогнозное значение системе для сохранения в хранилище данных.

В завершении цикла Детектор аномалий определяет, завершает ли текущее значение датчика некоторую аномальную последовательность значений датчика, и уведомляет оператора системы в этом случае.

Далее перечисленные выше подсистемы Модуля очистки данных рассмотрены более детально.

3.2. Препроцессор

Препроцессор выполняет подготовку обучающей выборки для нейронной сети подсистемы Предиктор и состоит из следующих основных подсистем (см. рис. 3): Парсер, Восстановитель, Детектор выбросов и Нормализатор. Технологический цикл работы Препроцессора выглядит следующим образом.

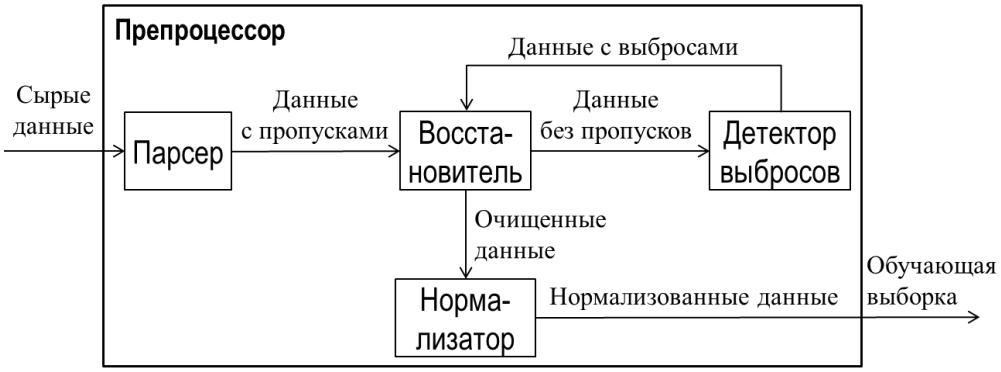


Рис. 3. Архитектура Препроцессора

Сначала Парсер извлекает данные датчика из хранилища данных и преобразует их в формат, пригодный для последующей обработки. Затем Восстановитель заменяет пропущенные значения на правдоподобные синтетические. После этого Детектор выбросов находит в данных точки-выбросы и заменяет их пустыми значениями NULL. Полученные данные повторно подаются на вход Восстановителя, который, таким образом, заменяет выбросы на правдоподобные синтетические значения. На последнем шаге Нормализатор формирует из полученных данных набор нормализованных подпоследовательностей, которые являются обучающей выборкой для нейронной сети подсистемы Предиктор.

Восстановитель использует сезонную авторегрессионную интегрированную модель скользящего среднего с эксогенными регрессорами *SARIMAX (Seasonal AutoRegressive Integrated Moving Average with exogenous regressors)* [10], которая часто применяется для прогнозирования значений сезонных временных рядов в различных предметных областях. Свойство сезонности временного ряда подразумевает наличие в данном ряде колебаний, возникающих с некоторой периодичностью. Для применения SARIMAX в данной предметной области необходимо подобрать следующие параметры: p — порядок авторегрессии, d — порядок интегрирования, q — порядок скользящего среднего, P — сезонный порядок авторегрессии, D — порядок сезонного интегрирования, Q — сезонный порядок скользящего среднего, s — период сезонности временного ряда. Применение SARIMAX возможно, если целевой временной ряд является стационарным. Под стационарными вре-

менными рядами понимают такие временные ряды, элементы которых являются случайными величинами с постоянным математическим ожиданием и постоянной дисперсией. Стационарность временного ряда проверяется с помощью следующих двух тестов, применяемых последовательно: расширенный тест Дики—Фуллера (ADF) [11] и тест Квятковского—Филлипса—Шмидта—Шина (KPSS) [21]. Если указанные тесты не подтвердят стационарность временного ряда, то для применения модели SARIMAX необходимо преобразовать элементы ряда в зависимости от его характеристик, например, логарифмировать элементы в случае наличия в ряде мультипликативной сезонности (существенного отличия дисперсии элементов в различных участках ряда).

В более формальном изложении работа Препроцессора выглядит следующим образом. Парсер извлекает из хранилища данных временной ряд T, который представляет собой хронологически упорядоченную последовательность числовых либо пустых значений: $T = (t_1, \ldots, t_m)$, где $t_i \in \mathbb{R}$ или $t_i = \mathrm{NULL}$ и число m называется длиной временного ряда. Затем тандем Восстановителя и Детектора выбросов преобразует T таким образом, чтобы $\forall i\ t_i \in \mathbb{R}$.

Далее Нормализатор подготавливает обучающую выборку, каждый элемент которой представляет собой пару «последовательность показаний датчика» и «прогнозное значение датчика». В качестве первого элемента указанной пары берется нормализованная подпоследовательность показаний датчика фиксированной длины, в качестве второго элемента — одно нормализованное показание датчика, следующее за данной подпоследовательностью.

Подпоследовательностью $T_{i,n}$ временного ряда T назовем непрерывный промежуток значений указанного ряда, состоящий из n элементов и начинающийся с позиции i: $T_{i,n} = (t_i, \ldots, t_{i+n-1})$, $1 \le i \le m - n + 1$. Используя минимаксную нормализацию, для подпоследовательности $T_{i,n}$ ее нормализованная версия $\tilde{T}_{i,n}$ вычисляется как $\tilde{T}_{i,n} = (\tilde{t}_i, \ldots, \tilde{t}_{i+n-1})$, где $\tilde{t}_i = \frac{t_i - t_{min}}{t_{max} - t_{min}}$. Для нормализованной подпоследовательности $\tilde{T}_{i,n}$ элемент $\tilde{t}_{n+1}$ рассматривается как ее прогнозное значение. Для дальнейшего использования введем обозначение S_T^n набора нормализованных подпоследовательностей временного ряда T, имеющих длину n $(n \ll m)$, и за P обозначим набор соответствующих этим подпоследовательностям прогнозов.

Длина подпоследовательности n $(n \ll m)$ является параметром Модуля очистки и вычисляется как $n = frequency \cdot horizon$, где $frequency$ — частота датчика, а $horizon$ — исторический горизонт (длина временного интервала в прошлом), используемый Предиктором и подбираемый оператором SCADA-системы «ПолиТЭР». Например, типичная частота снятия показаний температурного датчика 4 раза в час, исторический горизонт выбран равным 12 часам, тогда длина подпоследовательности, используемая Модулем очистки, составляет n=48.

3.3. Предиктор

Предиктор представляет собой рекуррентную нейронную сеть (Recurrent Neural Network, RNN) со слоем долгой краткосрочной памяти (Long Short-Term Memory, LSTM) [14]. Обучение рекуррентной нейронной сети выполняется на наборе данных, подготовленном Препроцессором. При использовании нейронная сеть принимает в качестве входных данных подпоследовательность реальных значений датчика, предшествующих текущему значению, и выдает прогнозное значение.

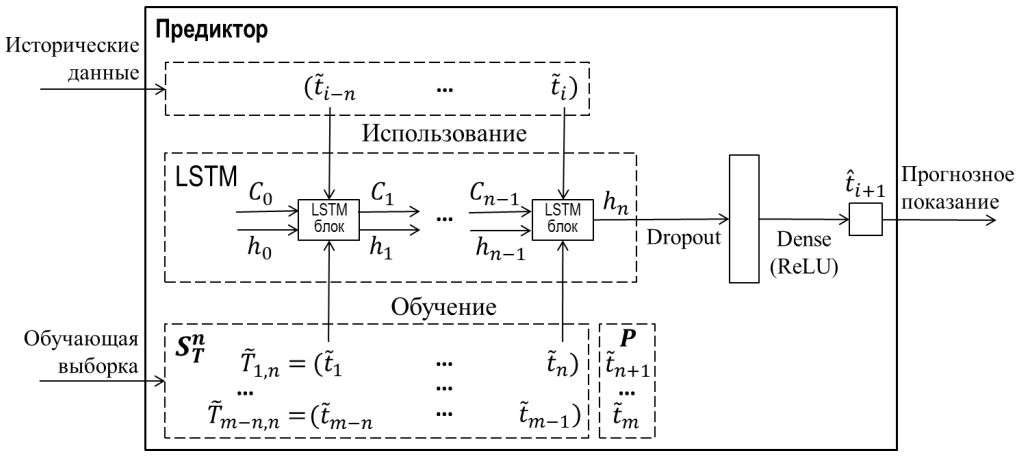


Рис. 4. Архитектура Предиктора

Архитектура Предиктора представлена на рис. 4. Рекуррентная нейронная сеть состоит из следующих слоев. Слой LSTM состоит из однотипных LSTM-блоков, где количество блоков равно длине подпоследовательности n, выбранной на этапе предварительной обработки. Работая вместе, блоки LSTM создают вектор-столбец h, т.н. скрытое состояние. Длина этого вектора — параметр, определяемый оператором SCADA-системы «ПолиТЭР». Слой Dropout случайным образом деактивирует предопределенную долю нейронов в векторе h для предотвращения переобучения нейронной сети. Доля деактивируемых нейронов также является параметром, который определяется оператором системы, с типичным значением 20 %. Слой Dense применяет выпрямленный линейный блок (Rectified Linear Unit, ReLU) в качестве функции активации для преобразования данных в одно прогнозное значение $\hat{t}_{i+1}$.

Каждый LSTM-блок состоит из состояния блока и нескольких слоев фильтра. Состояние блока — это вектор, который несет информацию из предыдущих моментов и будет проходить через всю цепочку LSTM-блоков. LSTM-блок имеет три слоя фильтра («вентили»), которые используются для контроля потоков информации на входах и на выходах памяти данных блоков: входной фильтр, фильтр забывания и выходной фильтр, которые регулируют объем данных, которые должны быть сохранены, забыты и доставлены на выход соответственно.

3.4. Реконструктор

Реконструктор получает на входе непустое текущее показание датчика и проверяет, существенно ли оно отличается от денормализованного синтетического значения, полученного с помощью Предиктора. Если это так, то реальное значение заменяется на синтетическое, которое в итоге сохраняется в хранилище данных.

Реализация Реконструктора основана на использовании распределения вероятностей ошибки прогнозирования [22]. В соответствии с этим подходом определяется порог различия $\varepsilon > 0$ между реальным показанием датчика t_{i+1} и прогнозным синтетическим значением датчика $\hat{t}_{i+1}$. Если $|t_{i+1} - \hat{t}_{i+1}| \geq \varepsilon$, то t_{i+1} представляет собой выброс. Порог различия вычисляется как $\varepsilon = \mu + k\sigma$, где μ — среднее значение ошибок прогнозирования, σ — стандартное отклонение ошибок прогнозирования, а $k > 0$ — параметр, определяемый оператором системы ПолиТЭР (с типичным значением k=3). Для некоторой подпоследовательности из тестовой выборки подсистемы Предиктор ошибка прогнозирования

вычисляется как модуль разности между последней точкой данной подпоследовательности и соответствующим прогнозным значением, которое было выдано подсистемой Предиктор.

3.5. Детектор аномалий

Детектор аномалий проверяет, является ли аномалией некая подпоследовательность показаний датчика, которая заканчивается его текущим значением, и в случае положительного ответа формирует уведомление для оператора системы ПолиТЭР. В качестве длины указанной подпоследовательности рассматривается набор значений, которые соответствуют значимым временным интервалам в сфере управления отоплением и вычисляются в зависимости от частоты снятия показаний датчика. Например, если частота датчика составляет 4 раза в час, и оператору системы ПолиТЭР необходимо уведомление об аномалиях за прошедшие 12 час., 24 час. и 48 час., то в качестве параметра Детектора аномалий оператор должен задать следующие длины подпоследовательностей: 48, 96 и 182 соответственно.

В реализации Детектора аномалий нами используется концепция диссонанса [18, 27]. Диссонанс, в отличие от большинства алгоритмов обнаружения аномалий, требует только один интуитивно понятный параметр: длина аномальной подпоследовательности [19] и формально определяется следующим образом. Две подпоследовательности $T_{i,n}$ и $T_{j,n}$ ряда T не являются тривиальными совпадениями друг друга, если $\exists T_{p,n} \in S_T^n, i < p < j$: $\mathrm{ED}(T_{i,n}, T_{j,n}) < \mathrm{ED}(T_{i,n}, T_{p,n})$, где $\mathrm{ED}(\cdot,\cdot)$ обозначает евклидово расстояние. Пусть M_C обозначает множество подпоследовательностей, которые не являются тривиальным совпадением подпоследовательности $C \in S_T^n$. Тогда подпоследовательность $D \in S_T^n$ назовем наиболее значимым диссонансом в T, если $\forall C \in S_T^n \min(\mathrm{ED}(D, M_D)) > \min(\mathrm{ED}(C, M_C))$.

Иными словами, подпоследовательность ряда является диссонансом, если евклидово расстояние от нее до ближайшей подпоследовательности, не являющейся ее тривиальным совпадением, является наибольшим. Степень значимости диссонансов определяется следующим образом. Подпоследовательность $D \in S_T^n$ называется наиболее значимым k-м диссонансом в T, если евклидово расстояние от нее до k-й ближайшей подпоследовательности, не являющейся ее тривиальным совпадением, является наибольшим.

Таким образом, Детектор аномалий формирует уведомление для оператора системы ПолиТЭР, если подпоследовательность предопределенной длины, заканчивающаяся текущим показанием датчика, является диссонансом со степенью значимости не ниже k, где k является параметром, также определяемым оператором системы.

3.6. Инструменты и технологии реализации

Препроцессор реализован с помощью библиотек языка Python следующим образом. Парсер использует стандартные библиотеки *openpyxl* и *pandas*. Детектор выбросов основан на алгоритмах библиотеки *adtk (Anomaly Detection Toolkit)* [7]. Модель SARIMAX, используемая Восстановителем, реализована в стандартной библиотеке *statsmodels*. Нормализатор реализован с помощью стандартной библиотеки *sklearn*.

Предиктор реализован на основе библиотеки *Keras* [20] и фреймворка *TensorFlow* [4]. Для обучения рекуррентной нейронной сети использовались подпоследовательности

длины n=48, соответствующие 12 час. работы температурного датчика. Скрытое состояние рекуррентной нейронной сети было взято как вектор-столбец длины $|h|$=32. Для обучения сети использовались следующие общепринятые параметры: функция потерь MSE (Mean Square Error, среднеквадратическая ошибка), оптимизатор обучения Adam, количество эпох и размер пакета — 15 и 32 соответственно.

Для реализации Детектора аномалий использована библиотека *MatrixProfile* для языка программирования Python [9].

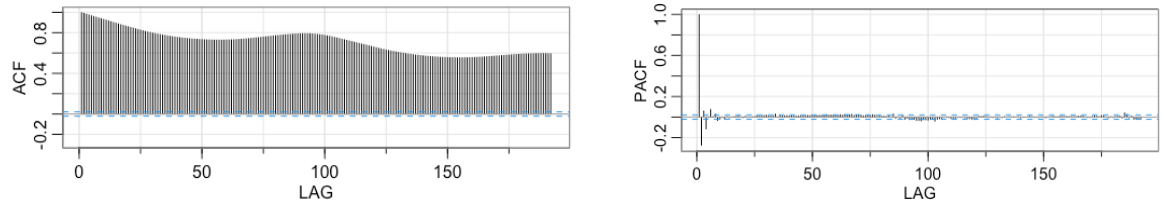
4. Вычислительные эксперименты

4.1. Набор данных

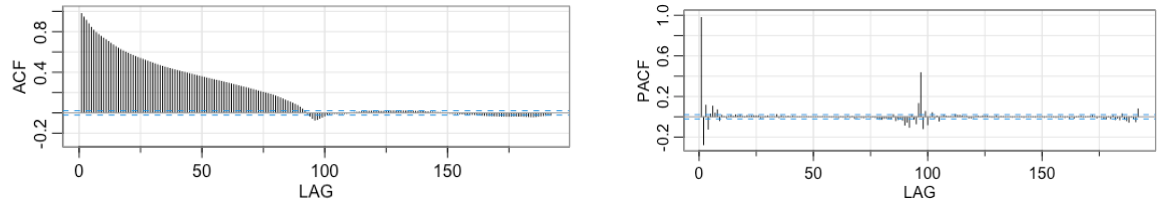
Для оценки предложенного подхода нами были проведены вычислительные эксперименты на реальных данных, взятых из хранилища данных SCADA-системы «ПолиТЭР». В качестве данных использовались показания температурного датчика, установленного в аудитории для потоковых лекций, за 2018 г. (частота снятия показаний датчика составляет 4 раза в час).

После обработки указанных данных Препроцессором первые 42 недели года (80 %) показаний были использованы в двух качествах: как выборка для подбора параметров модели SARIMAX Восстановителя и как обучающая выборка рекуррентной нейронной сети подсистемы Предиктор. Оставшиеся 8 недель года (20 %) показаний датчика моделировали работу модуля очистки данных в штатном режиме и использовались в качестве тестовой выборки.

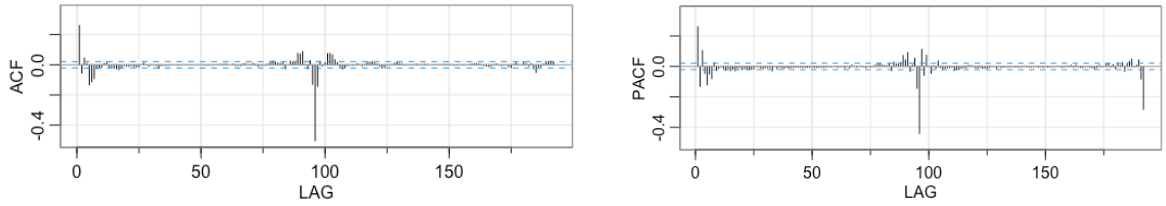
4.2. Подбор параметров модели SARIMAX



а) графики ACF и PACF для целевого временного ряда



б) графики ACF и PACF для ряда сезонных разностей



в) графики ACF и PACF для ряда первых разностей

Рис. 5. Коррелограммы исходного набора данных

Проверка взятого для исследования набора данных с помощью тестов ADF и KPSS подтвердила его нестационарность. Для приведения временного ряда к стационарному виду нами были последовательно применены техники вычитания первых разностей для удаления тренда и вычитания сезонных разностей для удаления сезонности [10]. Соответствующие графики функций автокорреляции (ACF) и частичной автокорреляции (PACF) приведены на рис. 5. Итоговые графики на рис. 5в позволяют подобрать следующие диапазоны для параметров модели SARIMAX: порядок авторегрессии $p \in [0, 7]$, порядок интегрирования $d = 1$, порядок скользящего среднего $q \in [0, 7]$, сезонный порядок авторегрессии $P = 1$, порядок сезонного интегрирования $D = 1$, сезонный порядок скользящего среднего $Q = 0$, период сезонности временного ряда $s = 96$.

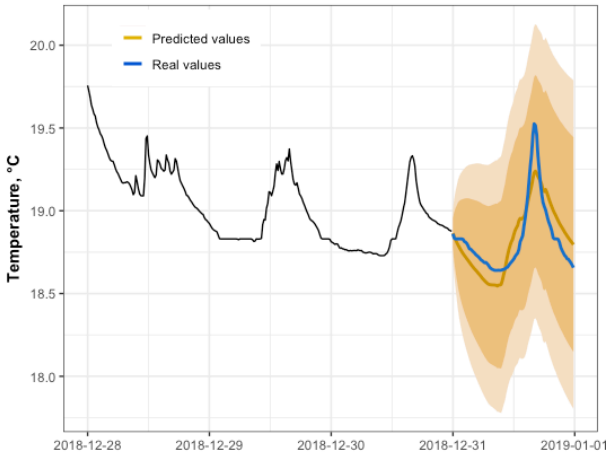
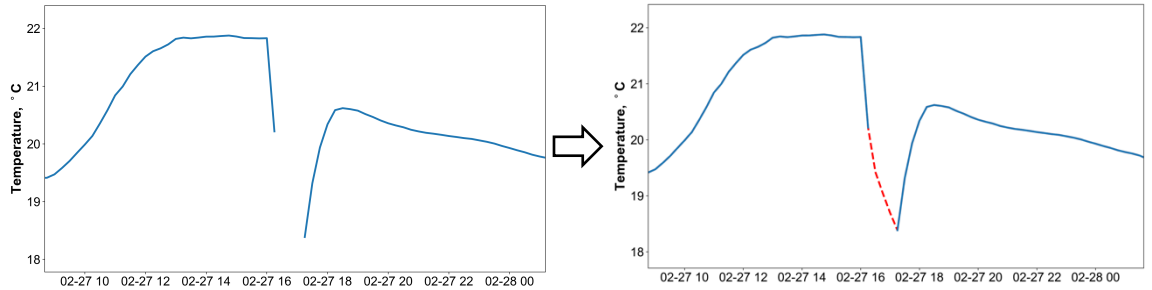
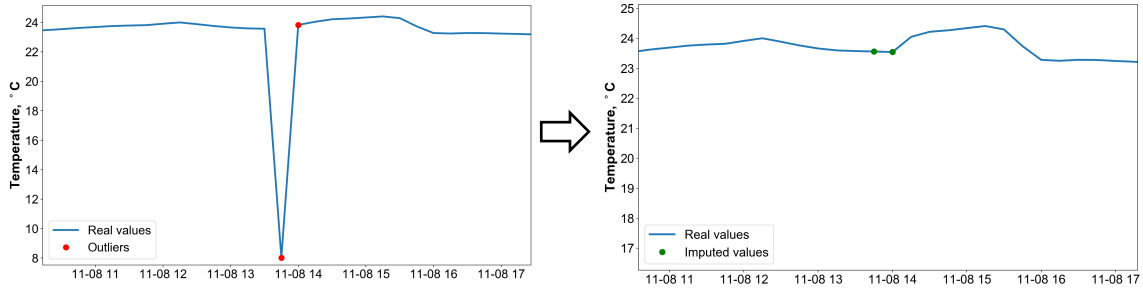


Рис. 6. Прогноз на один день, полученный с помощью модели SARIMAX(6,1,5)(0,1,1)$_{96}$



а) Восстановление пропущенных значений временного ряда



б) Замена выбросов временного ряда на правдоподобные синтетические значения

Рис. 7. Результаты работы Восстановителя на основе модели SARIMAX(6,1,5)(0,1,1)$_{96}$

Среди всех полученных моделей для штатной работы Модуля очистки данных была выбрана модель с параметрами $SARIMAX(p = 6, d = 1, q = 5)(P = 0, D = 1, Q = 1)s = 96$, поскольку имеет наименьшие значения информационных критериев Акаике (AIC) [6] и Шварца (BIC) [23] по сравнению с другими моделями и ряд остатков (разность между

фактическими и прогнозными значениями) обладает свойствами несмещенности, стационарности и неавтокоррелированности. На рис. 6 представлен прогноз на один день, а также построены 80 % и 95 % доверительные интервалы прогноза. Ошибка прогноза составила по мере $RMSE = 0.119$.

Поскольку подобранная нами модель SARIMAX довольно точно предсказывает значения температурного датчика, следовательно, она может быть использована для замены пропущенных значений на правдоподобные синтетические значения. Примеры работы Восстановителя представлены на рис. 7.

4.3. Оценка точности восстановления данных

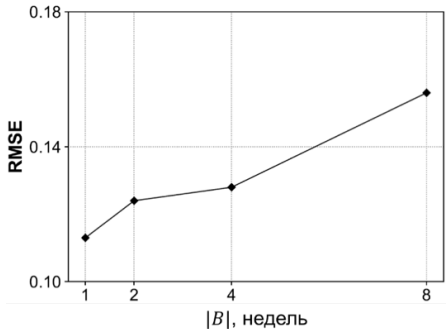
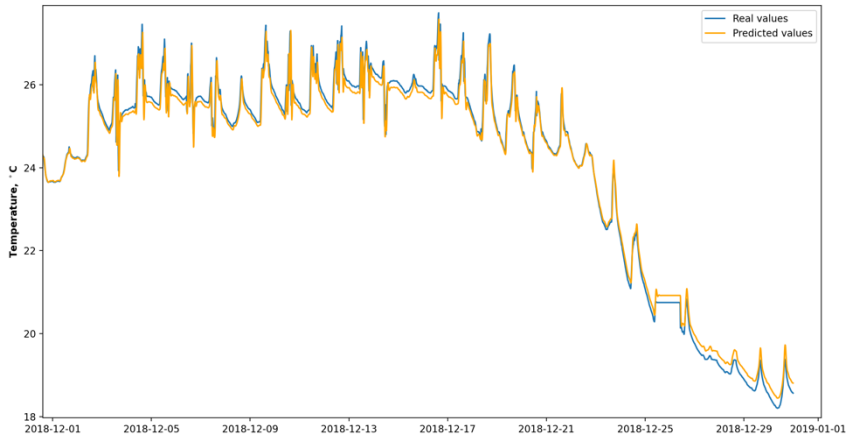
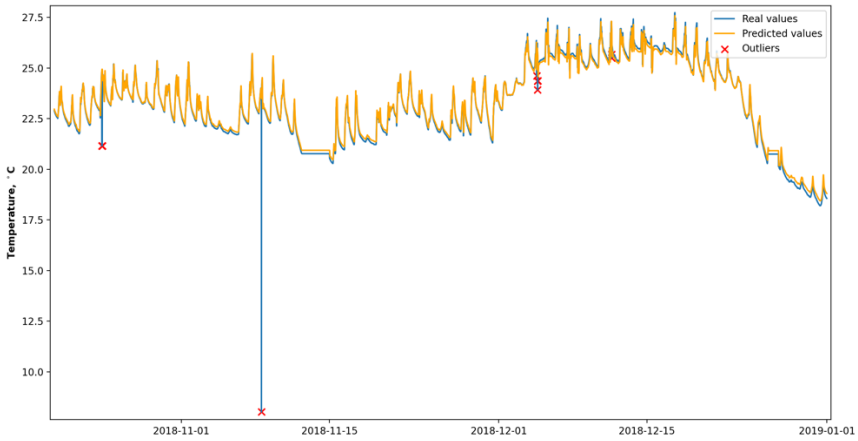


Рис. 8. Точность работы Модуля очистки данных



а) $|B| = 4$ недели



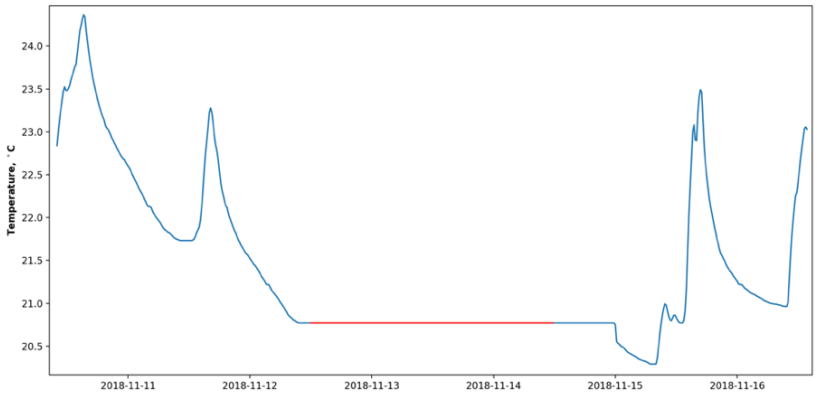
б) $|B| = 8$ недель

Рис. 9. Моделирование работы Модуля очистки данных

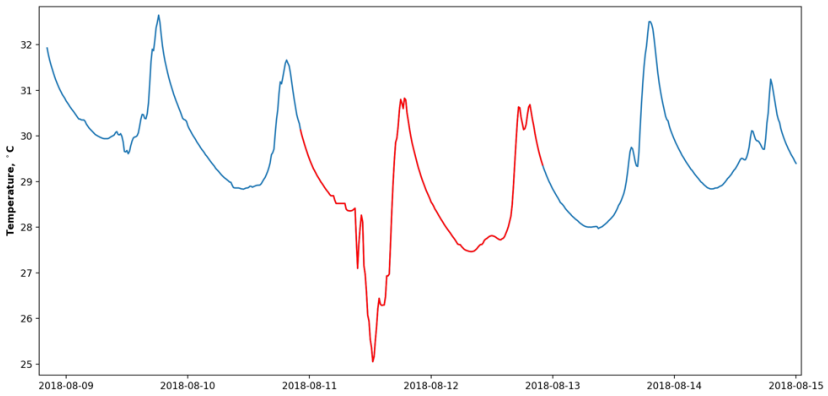
В экспериментах показания указанного датчика, синтезированные модулем очистки данных, сравнивались с реальными, полученными из хранилища данных, и оценивалась точность соответствия. В качестве меры точности использовалась среднеквадратичная ошибка (RMSE, Root Mean Square Error), определяемая следующим образом: $RMSE = \sqrt{\frac{1}{|B|}\sum_{i=1}^{|B|}(t_i - \hat{t}_i)^2}$, где t_i и $\hat{t}_i$ — реальное и синтетическое значения датчика соответственно, а $|B|$ обозначает длину блока показаний датчика, на которых выполнялось сравнение. Результаты экспериментов, в которых длина блока варьировалась от одной недели до двух месяцев, представлены на рис. 8. Как видно, разработанный модуль обеспечивает относительно высокую и стабильную точность.

На рис. 9 показаны примеры моделирования работы модуля для различных длин блока показаний датчика. Можно видеть, что модуль адекватно предсказывает нормальные значения и обнаруживает точечные выбросы в данных.

4.4. Обнаружение аномалий в данных



а) Топ-1 аномалия (активность датчика за 2 дня)



б) Топ-10 аномалия (2-дневная активность датчика)

Рис. 10. Аномалии, обнаруженные при моделировании работы Модуля очистки данных

На рис. 10 показан пример двух аномалий, обнаруженных при моделировании работы модуля очистки данных и соответствующих двухдневной активности вышеуказанного датчика. Первая аномалия представляет собой топ 1 диссонанс, найденный в тестовом наборе данных, которая может указывать на временный сбой в работе датчика. Вторая аномалия представляет собой топ 10 диссонанс и может указывать на быстрое снижение

температуры в лекционном зале из-за интенсивной вентиляции ввиду большого количества открытых окон в жаркий день. В любом случае, обнаруженные аномалии являются предметом реакции оператора.

Заключение

В статье затронута проблема очистки данных, поступающих с температурных датчиков в интеллектуальных системах отопления умных домов. Рассмотрен случай SCADA-системы ПолиТЭР, установленной в Южно-Уральском государственном университете (Челябинск) для управления отоплением университетского кампуса. Представлены архитектура и принципы реализации модуля очистки данных, разработанного и внедренного авторами в систему ПолиТЭР. Модуль очистки данных тиражируется для каждого температурного датчика системы ПолиТЭР и состоит из следующих подсистем: Препроцессор, Предиктор, Реконструктор и Детектор аномалий.

Препроцессор подготавливает обучающий набор данных для рекуррентной нейронной сети подсистемы Предиктор и реализуется с помощью следующих подсистем. Парсер извлекает данные датчика из хранилища данных и преобразует их для последующей обработки. Затем Восстановитель заменяет пропущенные значения на правдоподобные синтетические. После этого Детектор выбросов находит в данных точки-выбросы и заменяет их пустыми значениями, и полученные данные повторно обрабатываются Восстановителем. На последнем шаге Нормализатор формирует из полученных данных набор нормализованных подпоследовательностей, которые являются обучающей выборкой для рекуррентной нейронной сети подсистемы Предиктор.

Предиктор обеспечивает рекуррентную нейронную сеть долгой краткосрочной памяти, которая обучается на данных, подготовленных Препроцессором, получает на входе подпоследовательность реальных значений датчика, оканчивающихся его текущим значением, и выдает прогнозное значение датчика. Реконструктор получает на входе текущее значение датчика и проверяет, существенно ли оно отличается от синтетического значения, полученного с помощью Предиктора. Если это так, то реальное значение заменяется на синтетическое, которое в итоге сохраняется в хранилище данных. Детектор аномалий проверяет, является ли аномалией подпоследовательность показаний датчика, оканчивающаяся его текущим значением, и в случае положительного ответа формирует уведомление для оператора системы ПолиТЭР.

Проведены вычислительные эксперименты на реальных данных датчиков системы ПолиТЭР, показавшие высокую точность работы Модуля очистки данных.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 20-07-00140) и Министерства образования и науки РФ (государственное задание FENU-2020-0022).

Литература

1. Басалаев А.А. Автоматизированный энергоменеджмент теплоэнергетического комплекса университетского городка // Вестник ЮУрГУ. Серия «Компьютерные технологии, управление, радиоэлектроника». 2015. Т. 15, № 4. С. 26–32. DOI: 10.14529/ctcr150403.

2. Нетбай Г.В., Онискив В.Д., Столбов В.Ю., Каримов Р.Р. Прогнозное управление локальной городской системой теплоснабжения на основе нейросетевого моделирования // Вестник ЮУрГУ. Серия «Компьютерные технологии, управление, радиоэлектроника». 2019. № 3(47). С. 28–38. DOI: 10.14529/ctcr200303.
3. Ясир Ш., Кравец А.Г., Анохин А.О., Пивоваров В.В., Астанков А.А. Сбор и анализ гетерогенных данных в управлении услугами ЖКХ по водоснабжению и водоотведению // Прикаспийский журнал: управление и высокие технологии. 2015. Т. 15, № 4. С. 26–32. DOI: 10.14529/ctcr150403.
4. Abadi M., Barham P., Chen J., et al. TensorFlow: A System for Large-Scale Machine Learning // Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation, OSDI'16 (Berkeley, CA, United States, November 2016). 2016. P. 265–283. URL: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.
5. Ahmad T., Chen H., Huanga Y. Short-Term Energy Prediction for District-Level Load Management Using Machine Learning Based Approaches // Energy Procedia. 2019. Vol. 158. P. 3331–3338. DOI: 10.1016/j.egypro.2019.01.967.
6. Akaike H. A new look at the statistical model identification // IEEE Transactions on Automatic Control. 1974. Vol. 19, no. 6. P. 716–723. DOI: 10.1109/TAC.1974.1100705.
7. Anomaly Detection Toolkit, User Guide. URL: https://arundo-adtk.readthedocs-hosted.com/en/stable/userguide.html (дата обращения: 02.09.2020).
8. Basalaev A., Tochilkin M., Shnayder D. Enhancing room thermal comfort conditions modeling in buildings through schedule-based indicator functions for possible variable thermal perturbation inputs // Proceedings of 2019 International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2019 (Chelyabinsk, Russia, March 25–29, 2019). 2019. P. 1–8. DOI: 10.1109/ICIEAM.2019.8742907.
9. Benschoten A.H.V., Ouyang A., Bischoff F., Marrs T. MPA: a novel cross-language API for time series analysis // Journal of Open Source Software. 2020. Vol. 5, no. 49. DOI: 10.21105/joss.02179.
10. Box G.E.P., Jenkins G.M., Reinsel G.C., Ljung G.M. Time Series Analysis: Forecasting and Control. Whiley, 2015. 712 p.
11. Dickey D.A., Fuller W.A. Distribution of the estimators for autoregressive time series with a unit root // Journal of the American Statistical Association. 1979. Vol. 74. P. 427–431.
12. Dunia R., Joe Qin S. Joint diagnosis of process and sensor faults using principal component analysis // Control Engineering Practice. 1998. Vol. 6, no. 4. P. 457–469. DOI: 10.1016/S0967-0661(98)00027-6.
13. Farouq Sh., Byttner S., Bouguelia M.-R., et al. Large-scale monitoring of operationally diverse district heating substations: A reference-group based approach // Engineering Applications of Artificial Intelligence. 2020. Vol. 90. P. 1–16. DOI: 10.1016/j.engappai.2020.103492.
14. Hochreiter S., Schmidhuber J. Long Short-Term Memory // Neural Computation. 1997. Vol. 9, no. 8. P. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
15. Hu Y., Chen H., Li G., et al. A statistical training data cleaning strategy for the PCA-based chiller sensor fault detection, diagnosis and data reconstruction method // Energy and Buildings. 2016. Vol. 112. P. 270–278. DOI: 10.1016/j.enbuild.2015.11.066.
16. Idowu S., Saguna S., Åhlund C., Schelén O. Applied machine learning: Forecasting heat load in district heating system // Energy and Buildings. 2016. Vol. 133. P. 478–488. DOI: 10.1016/j.enbuild.2016.09.068.

17. Jha K. Minimal loop extraction for leak detection in water pipe network // Proceedings of 2012 1st International Conference on Recent Advances in Information Technology, RAIT 2012 (Dhanbad, India, March 15–17, 2012). IEEE Computer Society, 2012. P. 687–693. DOI: 10.1109/RAIT.2012.6194578.
18. Keogh E.J., Lin J., Fu A.W. HOT SAX: efficiently finding the most unusual time series subsequence // Proceedings of the 5th IEEE International Conference on Data Mining, ICDM 2005 (Houston, Texas, USA, November 27–30, 2005). IEEE Computer Society, 2005. P. 226–233. DOI: 10.1109/ICDM.2005.79.
19. Keogh E.J., Lonardi S., Ratanamahatana C.A. Towards parameter-free data mining // Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Seattle, Washington, USA, August 22–25, 2004). ACM, 2004. P. 206–215. DOI: 10.1145/1014052.1014077.
20. Keras Developer Guides. URL: https://keras.io/guides/ (дата обращения: 02.09.2020).
21. Kwiatkowski D., Phillips P.C.B., Schmidt P., Shin Y. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? // Journal of Econometrics. 1992. Vol. 54, no. 1. P. 154–178. DOI: 10.1016/0304-4076(92)90104-Y.
22. Malhotra P., Vig L., Shroff G.M., Agarwal P. Long Short Term Memory Networks for anomaly detection in time series // Proceedings of the 23rd European Symposium on Artificial Neural Networks, ESANN 2015 (Bruges, Belgium, April 22–24, 2015). 2015. P. 89–94. URL: http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf.
23. Schwarz G. Estimating the dimension of a model // The Annals of Statistics. 1978. Vol. 6, no. 2. P. 461–464. DOI: 10.1214/aos/1176344136.
24. Turner W.J.N., Staino A., Basu B. Residential HVAC fault detection using a system identification approach // Energy and Buildings. 2017. Vol. 151. P. 1–17. DOI: 10.1016/j.enbuild.2017.06.008.
25. Venkatasubramanian V. Process Fault Detection and Diagnosis: Past, Present and Future // IFAC Proceedings Volumes. 2001. Vol. 34, no. 27. P. 1–13. DOI: 10.1016/S1474-6670(17)33563-2.
26. Yan K., Ji Zh., Shen W. Online fault detection methods for chillers combining extended Kalman filter and Recursive One-class SVM // Neurocomputing. 2017. Vol. 228. P. 205–212. DOI: 10.1016/j.neucom.2016.09.076.
27. Yankov D., Keogh E.J., Rebbapragada U. Disk aware discord discovery: finding unusual time series in terabyte sized datasets // Knowledge and Information Systems. 2008. Vol. 17, no. 2. P. 241–262. DOI: 10.1007/s10115-008-0131-9.
28. Zhao Y., Li T., Zhang X., Zhang C. Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future // Renewable and Sustainable Energy Reviews. 2019. Vol. 109. P. 85–101. DOI: 10.1016/j.rser.2019.04.021.
29. Zhao Y., Zhang C., Zhang Y., et al. A review of data mining technologies in building energy systems: Load prediction, pattern identification, fault detection and diagnosis // Energy and Built Environment. 2020. Vol. 1, no. 2. P. 149–164. DOI: 10.1016/j.enbenv.2019.11.003.
30. Zimmerman N., Dahlquist E., Kyprianidis K. Towards On-line Fault Detection and Diagnostics in District Heating Systems // Energy Procedia. 2017. Vol. 105. P. 1960–1966. DOI: 10.1016/j.egypro.2017.03.567.
31. Zymbler M., Kraeva Ya., Latypova E., Kumar S., Shnayder D., Basalaev A. Cleaning Sensor Data in Smart Heating Control System // Proceedings — 2020 Global Smart Industry Conference, GloSIC 2020 (Chelyabinsk, Russia, November 17–19, 2020). IEEE, 2020. P. 375–381. DOI: 10.1109/GloSIC50886.2020.9267813.

Цымблер Михаил Леонидович, д.ф.-м.н., доцент, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Краева Яна Александровна, преподаватель, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Латыпова Елизавета Альбертовна, студент, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Иванова Елена Владимировна, к.ф.-м.н., кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Шнайдер Дмитрий Александрович, д.т.н., доцент, кафедра «Автоматика и управление», Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Басалаев Александр Анатольевич, к.т.н., кафедра «Автоматика и управление», Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

DOI: 10.14529/cmse210302

CLEANING SENSOR DATA IN INTELLIGENT HEATING CONTROL SYSTEM

© 2021 M.L. Zymbler, Ya.A. Kraeva, E.A. Latypova, E.V. Ivanova, D.A. Shnayder, A.A. Basalaev

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia)

E-mail: mzym@susu.ru, kraevaya@susu.ru, latypovaea@susu.ru, elena.ivanova@susu.ru,shnaiderda@susu.ru, basalaevaa@susu.ru

Received: 03.09.2020

Sometimes, smart heating control applications are partially equipped with missing values and outliers in the sensor data due to software/hardware failures/human errors. To provide an effective analysis and decision-making, erroneous sensor data should be cleaned by imputation of missing values and smoothing outliers. In this paper, we present a case of the Smart Heating Control System (SHCS) installed in the South Ural State University, and describe the structure and development principles of Data Cleaning Module (DCM) of the system. We implement DCM through data mining and neural network technologies as a set of the following subsystems. The preprocessor extracts raw data from the system's data warehouse and prepares a training data for further processing. Predictor provides Recurrent Neural Network (RNN) to forecast the next value of a sensor based on its historical data. Reconstructor determines if the current value of a sensor is an outlier, and if so, imputes it by the synthetic value from Predictor. Finally, Anomaly Detector subsystem discovers anomalous sequences in the sensor data. In the experiments on the real sensor data, DCM showed relatively high and stable accuracy as well as adequate detection of anomalies.

Keywords: heating systems, control systems, cleaning, process control, temperature sensors, flowmeters, temperature measurement.

FOR CITATION

Zymbler M.L., Kraeva Ya.A., Latypova E.A., Ivanova E.V., Shnayder D.A., Basalaev A.A. Cleaning Sensor Data in Intelligent Heating Control System. *Bulletin of the* South Ural State University. Series: Computational Mathematics and Software Engineering. 2021. Vol. 10, no. 3. P. 16–36. (in Russian) DOI: 10.14529/cmse210302.

References

1. Basalaev A. Automated energy management for heat and power system of university campus. Bulletin of the South Ural State University. Ser. Computer Technologies, Automatic Control, Radio Electronics. 2015. Vol. 15, no. 4. P. 26–32. DOI: 10.14529/ctcr150403.
2. Netbay G.V, Oniskiv V.D., Stolbov V.Yu., Karimov R.R. Managdment of a local urban heat supply system based on neural network modeling taking into account the weather forecast. Bulletin of the South Ural State University. Ser. Computer Technologies, Automatic Control, Radio Electronics. 2019. Vol. 20, no. 3. P. 28–38. DOI: 10.14529/ctcr200303.
3. Yasir Sh., Kravets A.G., Anokhin A.O., Pivovarov V.V., Astankov A.A. Collection and analysis of heterogeneous data in the management of housing services on water supply and water disposal. Caspian Journal: Control and High Technologies. 2019. Vol. 3, no. 47. P. 26–32. DOI: 10.14529/ctcr150403.
4. Abadi M., Barham P., Chen J., et al. TensorFlow: A System for Large-Scale Machine Learning. Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation, OSDI'16 (Berkeley, CA, United States, November 2016). 2016. P. 265–283. URL: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.
5. Ahmad T., Chen H., Huanga Y. Short-Term Energy Prediction for District-Level Load Management Using Machine Learning Based Approaches. Energy Procedia. 2019. Vol. 158. P. 3331–3338. DOI: 10.1016/j.egypro.2019.01.967.
6. Akaike H. A new look at the statistical model identification. IEEE Transactions on Automatic Control. 1974. Vol. 19, no. 6. P. 716–723. DOI: 10.1109/TAC.1974.1100705.
7. Anomaly Detection Toolkit, User Guide. URL: https://arundo-adtk.readthedocs-hosted.com/en/stable/userguide.html (accessed: 02.09.2020).
8. Basalaev A., Tochilkin M., Shnayder D. Enhancing room thermal comfort conditions modeling in buildings through schedule-based indicator functions for possible variable thermal perturbation inputs. Proceedings of 2019 International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2019 (Chelyabinsk, Russia, March 25–29, 2019). 2019. P. 1–8. DOI: 10.1109/ICIEAM.2019.8742907.
9. Benschoten A.H.V., Ouyang A., Bischoff F., Marrs T. MPA: a novel cross-language API for time series analysis. Journal of Open Source Software. 2020. Vol. 5, no. 49. DOI: 10.21105/joss.02179.
10. Box G.E.P., Jenkins G.M., Reinsel G.C., Ljung G.M. Time Series Analysis: Forecasting and Control. Whiley, 2015. 712 p.
11. Dickey D.A., Fuller W.A. Distribution of the estimators for autoregressive time series with a unit root. Journal of the American Statistical Association. 1979. Vol. 74. P. 427–431.
12. Dunia R., Joe Qin S. Joint diagnosis of process and sensor faults using principal component analysis. Control Engineering Practice. 1998. Vol. 6, no. 4. P. 457–469. DOI: 10.1016/S0967-0661(98)00027-6.
13. Farouq Sh., Byttner S., Bouguelia M.-R., et al. Large-scale monitoring of operationally diverse district heating substations: A reference-group based approach. Engineering Applications of Artificial Intelligence. 2020. Vol. 90. P. 1–16. DOI: 10.1016/j.engappai.2020.103492.

14. Hochreiter S., Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997. Vol. 9, no. 8. P. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
15. Hu Y., Chen H., Li G., et al. A statistical training data cleaning strategy for the PCA-based chiller sensor fault detection, diagnosis and data reconstruction method. Energy and Buildings. 2016. Vol. 112. P. 270–278. DOI: 10.1016/j.enbuild.2015.11.066.
16. Idowu S., Saguna S., Åhlund C., Schelén O. Applied machine learning: Forecasting heat load in district heating system. Energy and Buildings. 2016. Vol. 133. P. 478–488. DOI: 10.1016/j.enbuild.2016.09.068.
17. Jha K. Minimal loop extraction for leak detection in water pipe network. Proceedings of 2012 1st International Conference on Recent Advances in Information Technology, RAIT 2012 (Dhanbad, India, March 15–17, 2012). IEEE Computer Society, 2012. P. 687–693. DOI: 10.1109/RAIT.2012.6194578.
18. Keogh E.J., Lin J., Fu A.W. HOT SAX: efficiently finding the most unusual time series subsequence. Proceedings of the 5th IEEE International Conference on Data Mining, ICDM 2005 (Houston, Texas, USA, November 27–30, 2005). IEEE Computer Society, 2005. P. 226–233. DOI: 10.1109/ICDM.2005.79.
19. Keogh E.J., Lonardi S., Ratanamahatana C.A. Towards parameter-free data mining. Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Seattle, Washington, USA, August 22–25, 2004). ACM, 2004. P. 206–215. DOI: 10.1145/1014052.1014077.
20. Keras Developer Guides. URL: https://keras.io/guides/ (accessed: 02.09.2020).
21. Kwiatkowski D., Phillips P.C.B., Schmidt P., Shin Y. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? Journal of Econometrics. 1992. Vol. 54, no. 1. P. 154–178. DOI: 10.1016/0304-4076(92)90104-Y.
22. Malhotra P., Vig L., Shroff G.M., Agarwal P. Long Short Term Memory Networks for anomaly detection in time series. Proceedings of the 23rd European Symposium on Artificial Neural Networks, ESANN 2015 (Bruges, Belgium, April 22–24, 2015). 2015. P. 89–94. URL: http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf.
23. Schwarz G. Estimating the dimension of a model. The Annals of Statistics. 1978. Vol. 6, no. 2. P. 461–464. DOI: 10.1214/aos/1176344136.
24. Turner W.J.N., Staino A., Basu B. Residential HVAC fault detection using a system identification approach. Energy and Buildings. 2017. Vol. 151. P. 1–17. DOI: 10.1016/j.enbuild.2017.06.008.
25. Venkatasubramanian V. Process Fault Detection and Diagnosis: Past, Present and Future. IFAC Proceedings Volumes. 2001. Vol. 34, no. 27. P. 1–13. DOI: 10.1016/S1474-6670(17)33563-2.
26. Yan K., Ji Zh., Shen W. Online fault detection methods for chillers combining extended Kalman filter and Recursive One-class SVM. Neurocomputing. 2017. Vol. 228. P. 205–212. DOI: 10.1016/j.neucom.2016.09.076.
27. Yankov D., Keogh E.J., Rebbapragada U. Disk aware discord discovery: finding unusual time series in terabyte sized datasets. Knowledge and Information Systems. 2008. Vol. 17, no. 2. P. 241–262. DOI: 10.1007/s10115-008-0131-9.
28. Zhao Y., Li T., Zhang X., Zhang C. Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future. Renewable and Sustainable Energy Reviews. 2019. Vol. 109. P. 85–101. DOI: 10.1016/j.rser.2019.04.021.

29. Zhao Y., Zhang C., Zhang Y., et al. A review of data mining technologies in building energy systems: Load prediction, pattern identification, fault detection and diagnosis. Energy and Built Environment. 2020. Vol. 1, no. 2. P. 149–164. DOI: 10.1016/j.enbenv.2019.11.003.
30. Zimmerman N., Dahlquist E., Kyprianidis K. Towards On-line Fault Detection and Diagnostics in District Heating Systems. Energy Procedia. 2017. Vol. 105. P. 1960–1966. DOI: 10.1016/j.egypro.2017.03.567.
31. Zymbler M., Kraeva Ya., Latypova E., Kumar S., Shnayder D., Basalaev A. Cleaning Sensor Data in Smart Heating Control System. Proceedings — 2020 Global Smart Industry Conference, GloSIC 2020 (Chelyabinsk, Russia, November 17–19, 2020). IEEE, 2020. P. 375–381. DOI: 10.1109/GloSIC50886.2020.9267813.

УДК 519.67 **DOI: 10.14529/cmse210303**

ПОДХОД К ОЦЕНКЕ ЛОКАЛЬНОСТИ ЗЕРНИСТЫХ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ, ЛОГИЧЕСКИ ОРГАНИЗОВАННЫХ В ДВУМЕРНУЮ СТРУКТУРУ*

© 2021 А.А. Толстиков[1], С.В. Баханович[2], Н.А. Лиходед[1]

[1]*Белорусский государственный университет*
(220030 Республика Беларусь, Минск, пр. Независимости, д. 4),
[2]*Институт математики НАН Беларуси*
(220072 Республика Беларусь, Минск, ул. Сурганова, д. 11)
E-mail: tolstikov@bsu.by, bsv@im.bas-net.by, likhoded@bsu.by

Поступила в редакцию: 31.07.2021

При реализации алгоритмов на многопроцессорных вычислительных устройствах важнейшую роль для достижения высокой производительности играет локальность — вычислительное свойство алгоритма, отражающее степень использования памяти с быстрым доступом. Для суперкомпьютеров с распределенной памятью быстрой считается локальная память вычислительного узла. Параллельные алгоритмы с меньшим объемом и лучшей структурой коммуникационных операций обладают лучшей локальностью. В работе на основе схемы расщепления с весами построен новый параллельный алгоритм численного решения трехмерного линейного уравнения теплопроводности. Алгоритм ориентирован на компьютеры с распределенной памятью, сочетает конвейерный и естественый параллелизм, использует 2D структуру процессов. Схема расщепления обладает естественным параллелизмом. Ранее для случая 1D структуры процессов было показано, что использование конвейерного параллелизма вместо части естественного параллелизма приводит к меньшим объемам и лучшей структуре коммуникационных операций. Построенный 2D алгоритм обобщает известный 1D алгоритм. Использование двумерных структур позволяет уменьшить объем и улучшить структуру коммуникационных операций, уменьшить время разгона и торможения вычислений. Поэтому 2D алгоритм обладает лучшей локальностью по сравнению с использованием 1D структуры процессов. Вычислительные эксперименты на суперкомпьютере показали преимущество нового параллельного алгоритма. По аналогии с представленным алгоритмом можно получить и исследовать параллельные алгоритмы для других схем метода дробных шагов. На примере алгоритма, реализующего схему расщепления, представлен подход к получению асимптотических оценок объема коммуникационных операций зернистых (т.е. уровня макроопераций) параллельных вычислительных процессов, логически организованных в двумерную структуру. Оценки могут быть использованы для сравнения коммуникационных затрат при получении альтернативных вариантов параллельных алгоритмов.

Ключевые слова: параллельные вычисления, многопроцессорная вычислительная система с распределенной памятью, уменьшение обменов данными, метод дробных шагов.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Толстиков А.А., Баханович С.В., Лиходед Н.А. Подход к оценке локальности зернистых вычислительных процессов, логически организованных в двумерную структуру // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 3. С. 37–55. DOI: 10.14529/cmse210303.

Введение

В качестве целевого компьютера для реализации алгоритмов будем рассматривать многопроцессорную вычислительную систему с распределенной памятью. При многопроцессорной обработке памятью с быстрым доступом считается локальная память процессора, при однопроцессорной — кэш-память. Локальность алгоритма — это вычислительное свойство алгоритма, отражающее степень использования памяти с быстрым доступом. Использова-

* Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии (ПаВТ) 2021».

ние локальности играет важнейшую роль для достижения высокой эффективности реализации алгоритмов на многопроцессорных вычислительных устройствах [4].

Параллельные алгоритмы для компьютеров с распределенной памятью должны быть зернистыми: множество операций алгоритма должно быть разбито на множества, называемые зернами вычислений. Можно использовать тайлинг (tiling) — преобразование алгоритма для получения макроопераций-тайлов [18, 20]. Операции одного тайла выполняются атомарно, как одна единица вычислений, а обмен данными происходит массивами.

Локальность параллельного алгоритма, предназначенного для реализации на компьютерах с распределенной памятью, характеризует коммуникационные затраты: чем меньше при заданном числе вычислительных ядер суммарный объем операций обмена данными, тем лучше локальность. Задачей исследования локальности параллельного алгоритма является оценка числа и объема коммуникационных операций. На локальность влияет также структура операций обмена данными: число и топология («близость») процессов, вовлеченных в групповые коммуникации. Хорошей локальности можно достичь путем преобразования алгоритмов и/или удачного распределения операций и данных между процессорами [1, 3, 8, 9, 16, 18, 20].

Целью этой работы является разработка и исследование локальности нового параллельного алгоритма, реализующего схему расщепления с весами численного решения трехмерного линейного уравнения теплопроводности. Трехшаговая схема расщепления [6, 15] является одной из наиболее употребительных схем метода дробных шагов численного решения трехмерных линейных и квазилинейных уравнений теплопроводности. Схема абсолютно устойчива, имеет второй порядок аппроксимации. К преимуществам схемы расщепления относится также простота по сравнению с другими схемами метода дробных шагов, также обладающими свойствами сильной устойчивости и второго порядка точности.

Предлагаемый алгоритм ориентирован на параллельные компьютеры с распределенной памятью, использует 2D структуру процессов и обладает лучшей локальностью по сравнению с известными алгоритмами [14], использующими 1D структуры процессов. Схема расщепления обладает естественным параллелизмом, но при реализации на параллельных компьютерах с распределенной памятью использование этого естественного параллелизма приводит к большим коммуникационным издержкам. Так же, как и 1D алгоритм, предложенный в [14], новый 2D алгоритм использует конвейерный параллелизм вместо части естественного параллелизма, что приводит к меньшим объемам и лучшей структуре коммуникационных операций. Сформулировано и обосновано утверждение, устанавливающее объем и структуру коммуникационных операций алгоритма. По аналогии с представленным алгоритмом можно получить и исследовать 2D параллельные алгоритмы и для других схем метода дробных шагов.

Фактически на примере алгоритма, реализующего схему расщепления, представлен подход к организации параллельных двумерных вычислительных процессов и получению асимптотических оценок объема коммуникационных операций. Подход использует анализ информационных зависимостей, порождающих коммуникационные операции, и основан на сформулированных ранее утверждениях для случая вычислительных процессов, логически организованных в одномерную структуру.

Статья структурирована следующим образом. В разделе 1 дан обзор работ по тематике исследования. В разделе 2 даны разностная схема расщепления численного решения трехмерного уравнения теплопроводности и псевдокод основной части алгоритма, реализующего рассмотренную схему, в котором указаны циклы, итерации которых заведомо можно выполнять независимо. В разделе 3 приведены необходимые для дальнейших исследований функции, задающие информационные связи между операциями рассмотренного алгоритма. В разделе 4 предложен способ получения 2D зернистых (т.е. уровня макроопераций-тайлов) вычислительных процессов, отличительной особенностью которого является предположение независимости первой и второй координат 2D процессов. В разделе 5 задано распределение операций алгоритма, реализующего рассмотренную схему расщепления, между 2D зернистыми процессами; это распределение операций приводит к конвейерному параллелизму вместо части естественного параллелизма. В разделе 6 оценивается объем и структура коммуникационных операций нового параллельного алгоритма, приведен его псевдокод. В разделе 7 обсуждаются результаты вычислительных экспериментов. В заключении суммируются основные результаты, намечаются направления дальнейших исследований.

1. Обзор работ

Как уже отмечалось, при реализации алгоритмов на многопроцессорных вычислительных устройствах для достижения высокой производительности важнейшую роль играет использование локальности. Отметим некоторые подходы к оптимизации использования иерархической памяти, уменьшению числа и объема коммуникационных операций при использовании многоядерных CPU, компьютеров с распределенной памятью, графических ускорителей (GPU).

Очевидным приемом является выявление независимых частей алгоритма [3, 8, 19]. В случае обнаружения таких частей все ядра параллельной вычислительной системы могут работать независимо друг от друга, коммуникационные операции не требуются. На практике, однако, алгоритм не часто допускает декомпозицию на независимые части. Еще один подход заключается в использовании макроопераций-тайлов (получение блочных версий алгоритмов) [8, 16, 17, 20]. Операции одного тайла выполняются атомарно, как одна единица вычислений. При использовании суперкомпьютеров с распределенной памятью обмен данными происходит массивами, тайлинг позволяет увеличить пакет передаваемых данных и уменьшить частоту коммуникаций, минимизировать накладные расходы на обмены данными. Разбиение множества операций на макрооперации-тайлы является очень полезным преобразованием также при реализации алгоритмов на многоядерных персональных компьютерах и на графических ускорителях. В работах [9, 18, 19] предлагаются методы получения аффинных преобразований, которые позволили бы уменьшить число информационных связей между множествами операций разбиваемого на части алгоритма. Операциям алгоритма назначается новый порядок выполнения, учитывающий параллелизм и локальность; такие преобразования приводят к уменьшению коммуникационных операций, так как операции обмена данными порождаются информационными связями.

В работах [2, 7, 21, 22] улучшение локальности значительно повышает скорость выполнения расчета при параллельной реализации сеточных задач.

В работе [11] для случая 1D процессов предложены и доказаны утверждения, позволяющие оценить коммуникационные затраты при реализации алгоритмов на параллельных компьютерах с распределенной памятью; получены выражения, характеризующие число данных, для которых требуются коммуникации, и число процессов, вовлеченных в пересылки этих данных. Использование двумерных структур связано, по сравнению с одномерными структурами, с дополнительными ограничениями при реализации алгоритмов на суперкомпьютерах с распределенной памятью: для возможности организовать полностью загруженные работой параллельные вычислительные процессы требуется произвести тайлинг по трем координатам многомерного итерационного пространства, в то время как для случая одномерной структуры достаточно произвести тайлинг по двум координатам. Но часто, как и в этой работе, случай использования 2D процессов имеет ряд преимуществ: позволяет уменьшить объем коммуникационных операций, уменьшить разгон и торможение вычислений, получить большее число вычислительных процессов (что важно, если число итераций по одной координате сравнительно небольшое) [10].

2. Вычислительный алгоритм, реализующий разностную схему расщепления

Рассмотрим в области $Q_T = G \times [0 < t \le T]$, $G = [0 < x_1 < l_1] \times [0 < x_2 < l_2] \times [0 < x_3 < l_3]$, трехмерное уравнение теплопроводности:

$$\frac{\partial u}{\partial t} = Lu + f(x_1, x_2, x_3, t), \;\; L = \sum_{m=1}^{3} L_m, \;\; L_m = k \frac{\partial^2 u}{\partial x_m^2}, \;\; k{=}\text{const}{>}0, \tag{1}$$

с начальными и краевыми условиями

$$u(x_1, x_2, x_3, 0) = u_0(x_1, x_2, x_3), \; (x_1, x_2, x_3) \in G, \tag{2}$$

$$u(x_1, x_2, x_3, t)\Big|_{S_t} = \mu(x_1, x_2, x_3, t), \quad (x_1, x_2, x_3, t) \in S_t \equiv \partial Q_T. \tag{3}$$

Введем на отрезке $[0{<}t{\le}T]$ сетку узлов $\omega_\tau = \{t_j = j\tau, j = 0, 1, ..., j_0, j_0\tau = T\}$, в области G и на ее границе введем сетку узлов $\overline{\omega}_h = \{x_1^{(i_1)} = i_1 h_1, \;\; i_1 = 0, 1, ..., N_1, \;\; N_1 h_1 = l_1, \;\; x_2^{(i_2)} = i_2 h_2, \;\; i_2 = 0, 1, ..., N_2, \;\; N_2 h_2 = l_2, \;\; x_3^{(i_3)} = i_3 h_3, \;\; i_3 = 0, 1, ..., N_3, \;\; N_3 h_3 = l_3\}$.

Обозначим $\Lambda_m y^j = \frac{y_{i+1}^j - 2y_i^j + y_{i-1}^j}{h_m^2}$, где индекс i соответствует узлам x_m, остальные индексы для простоты опущены; $\Lambda_m y^j$ аппроксимирует вторую производную по направлению x_m в точках временного слоя j ($t{=}j\tau$) со вторым порядком точности. Для численного решения задачи применим схему расщепления с весами [6, 15] (рассмотрен случай $f{=}0$, $k{=}1$, выбраны веса, равные 0.5):

$$\frac{y^{j+1/3} - y^j}{\tau} = \frac{1}{2}\Lambda_1\left(y^{j+1/3} + y^j\right), \;\; \frac{y^{j+2/3} - y^{j+1/3}}{\tau} = \frac{1}{2}\Lambda_2\left(y^{j+2/3} + y^{j+1/3}\right), \;\; \frac{y^{j+1} - y^{j+2/3}}{\tau} = \frac{1}{2}\Lambda_3\left(y^{j+1} + y^{j+2/3}\right).$$

Основную вычислительную часть алгоритма, реализующего рассмотренную трехшаговую разностную схему расщепления с весами, можно представить в виде следующего псевдокода (циклы, итерации которых заведомо можно выполнять независимо, запишем как dopar):

```
do j =1, j0
   dopar i2= 1, N2-1
      dopar i3= 1, N3-1
         do i1= 1, N1-1
S1:         alpha(i1+1)=F1(alpha(i1))
S2:         beta(i1+1)=F2(alpha(i1),beta(i1),y^j(i1,i2,i3),
                          y^j(i1-1,i2,i3),y^j(i1+1,i2,i3))
         do i1= 1, N1-1
S3:         y^{j+1/3}(N1-i1,i2,i3)=alpha(N1-i1+1)y^{j+1/3}(N1-i1+1,i2,i3)+beta(N1-i1+1)
   dopar i1= 1, N1-1
      dopar i3= 1, N3-1
         do i2= 1, N2-1
S4:         alpha(i2+1)=F3(alpha(i2))
S5:         beta(i2+1)=F4(alpha(i2),beta(i2),y^{j+1/3}(i1,i2,i3),
                          y^{j+1/3}(i1,i2-1,i3),y^{j+1/3}(i1,i2+1,i3))
         do i2= 1, N2-1
S6:         y^{j+2/3}(i1,N2-i2,i3)=alpha(N2-i2+1)y^{j+2/3}(i1,N2-i2+1,i3)+beta(N2-i2+1)
   dopar i1= 1, N1-1
      dopar i2= 1, N2-1
         do i3= 1, N3-1
S7:         alpha(i3+1)=F5(alpha(i3))
S8:         beta(i3+1)=F6(alpha(i3),beta(i3),y^{j+2/3}(i1,i2,i3),
                          y^{j+2/3}(i1,i2,i3-1),y^{j+2/3}(i1,i2,i3+1))
         do i3= 1, N3-1
S9:         y^{j+1}(i1,i2,N3-i3)=alpha(N3-i3+1)y^{j+1}(i1,i2,N3-i3+1)+beta(N3-i3+1)
```

Рис. 1. Алгоритм разностной схемы расщепления

Здесь $alpha(i)$ и $beta(i)$ — коэффициенты прогонки, возникающие при решении промежуточных систем линейных алгебраических уравнений. Конечные значения $y^j(i_1,i_2,i_3)$ являются приближенным решением задачи (1)–(3).

3. Информационная структура алгоритма расщепления с весами

Вхождением (a,S_β,q) будем называть q-е вхождение массива a в оператор S_β. Выполнение оператора S_β при конкретных значениях β и вектора параметров цикла J будем называть операцией и обозначать $S_\beta(J)$. Пара вхождений $(a,S_\alpha,1)$ и (a,S_β,q) порождает истинную зависимость $S_\alpha(I)\to S_\beta(J)$, если: $S_\alpha(I)$ выполняется раньше $S_\beta(J)$; $S_\alpha(I)$ переопределяет (изменяет) элемент массива a, а $S_\beta(J)$ использует этот элемент массива в качестве аргумента; между операциями $S_\alpha(I)$ и $S_\beta(J)$ этот элемент не переопределяется. Истинные зависимости и входные данные алгоритма порождают коммуникационные операции.

Зависимости между операциями можно задать функциями вида

$$\overline{\Phi}_{\alpha,\beta}(J)=\Phi_{\alpha,\beta}J+\Psi_{\alpha,\beta}N-\varphi^{\alpha,\beta},$$

$$J\in V_{\alpha,\beta},\ N\in\mathbb{Z}^s,\ \Phi_{\alpha,\beta}\in\mathbb{Z}^{n_\alpha\times n_\beta},\ \Psi_{\alpha,\beta}\in\mathbb{Z}^{n_\alpha\times s},\ \varphi^{\alpha,\beta}\in\mathbb{Z}^{n_\alpha},$$

где $N\in\mathbb{Z}^s$ — вектор внешних переменных алгоритма, s — число внешних переменных, n_α и n_β — глубина вложенности циклов, окружающих операторы S_α и S_β. Функция зависимостей $\overline{\Phi}_{\alpha,\beta}(J)$ позволяет для операции $S_\beta(J)$ найти операцию $S_\alpha(I)$, от которой $S_\beta(J)$ зависит. Функции зависимостей, называемые также покрывающими функциями графа ал-

горитма, являются удобным математическим аппаратом для описания информационных связей между операциями алгоритма [3].

Истинные зависимости представленного алгоритма, реализующего схему расщепления с весами, определяются следующими функциями зависимостей:

$\overline{\Phi}_{1,1}(j,\ i_2,\ i_3,\ i_1)=\overline{\Phi}_{1,2}(j,\ i_2,\ i_3,\ i_1)=\overline{\Phi}_{2,2}(j,\ i_2,\ i_3,\ i_1)=\overline{\Phi}_{3,3}(j,\ i_2,\ i_3,\ i_1)=(j,\ i_2,\ i_3,\ i_1{-}1)$,
$\overline{\Phi}_{1,3}(j,\ i_2,\ i_3,\ i_1)=\overline{\Phi}_{2,3}(j,\ i_2,\ i_3,\ i_1)=(j,\ i_2,\ i_3,\ N_1{-}i_1)$,
$\overline{\Phi}_{9,2}(j,\ i_2,\ i_3,\ i_1)=(j{-}1,\ i_1{-}\varphi,\ i_2,\ N_3{-}i_3)$, где φ равно 0, 1, –1,
$\overline{\Phi}_{4,4}(j,\ i_1,\ i_3,\ i_2)=\overline{\Phi}_{4,5}(j,\ i_1,\ i_3,\ i_2)=\overline{\Phi}_{5,5}(j,\ i_1,\ i_3,\ i_2)=\overline{\Phi}_{6,6}(j,\ i_1,\ i_3,\ i_2)=(j,\ i_1,\ i_3,\ i_2{-}1)$,
$\overline{\Phi}_{4,6}(j,\ i_1,\ i_3,\ i_2)=\overline{\Phi}_{5,6}(j,\ i_1,\ i_3,\ i_2)=(j,\ i_1,\ i_3,\ N_2{-}i_2)$,
$\overline{\Phi}_{3,5}(j,\ i_1,\ i_3,\ i_2)=(j,\ i_2{-}\varphi,\ i_3,\ N_1{-}i_1)$, где φ равно 0, 1, –1,
$\overline{\Phi}_{7,7}(j,\ i_1,\ i_2,\ i_3)=\overline{\Phi}_{7,8}(j,\ i_1,\ i_2,\ i_3)=\overline{\Phi}_{8,8}(j,\ i_1\ i_2,\ i_3)=\overline{\Phi}_{9,9}(j,\ i_1,\ i_2,\ i_3)=(j,\ i_1,\ i_2,\ i_3{-}1)$,
$\overline{\Phi}_{7,9}(j,\ i_1,\ i_2,\ i_3)=\overline{\Phi}_{8,9}(j,\ i_1\ i_2,\ i_3)=(j,\ i_1,\ i_2,\ N_3{-}i_3)$,
$\overline{\Phi}_{6,8}(j,\ i_1,\ i_2,\ i_3)=(j,\ i_1,\ i_3{+}\varphi,\ N_2{-}i_2)$, где φ равно 0, 1, –1.

Матрично-векторное представление некоторых из этих функций:

$$\overline{\Phi}_{1,1}(j,\ i_2,\ i_3,\ i_1)=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}\begin{pmatrix}j\\i_2\\i_3\\i_1\end{pmatrix}-\begin{pmatrix}0\\0\\0\\1\end{pmatrix},$$

$$\overline{\Phi}_{9,2}(j,\ i_2,\ i_3,\ i_1)=\begin{pmatrix}1&0&0&0\\0&0&0&1\\0&1&0&0\\0&0&-1&0\end{pmatrix}\begin{pmatrix}j\\i_2\\i_3\\i_1\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&1\end{pmatrix}\begin{pmatrix}j_0\\N_1\\N_2\\N_3\end{pmatrix}-\begin{pmatrix}1\\\varphi\\0\\0\end{pmatrix},$$

$$\overline{\Phi}_{4,6}(j,\ i_1,\ i_3,\ i_2)=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&-1\end{pmatrix}\begin{pmatrix}j\\i_1\\i_3\\i_2\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&1&0\end{pmatrix}\begin{pmatrix}j_0\\N_1\\N_2\\N_3\end{pmatrix},$$

$$\overline{\Phi}_{3,5}(j,\ i_1,\ i_3,\ i_2)=\begin{pmatrix}1&0&0&0\\0&0&0&1\\0&0&1&0\\0&-1&0&0\end{pmatrix}\begin{pmatrix}j\\i_1\\i_3\\i_2\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&1&0&0\end{pmatrix}\begin{pmatrix}j_0\\N_1\\N_2\\N_3\end{pmatrix}-\begin{pmatrix}0\\\varphi\\0\\0\end{pmatrix},$$

$$\overline{\Phi}_{6,8}(j,\ i_1,\ i_2,\ i_3)=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&-1&0\end{pmatrix}\begin{pmatrix}j\\i_1\\i_2\\i_3\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&1&0\end{pmatrix}\begin{pmatrix}j_0\\N_1\\N_2\\N_3\end{pmatrix}-\begin{pmatrix}0\\0\\\varphi\\0\end{pmatrix}.$$

Функции зависимостей можно найти исходя из определения зависимостей или методом работы [3].

4. Организация и подход к оценке локальности 2D зернистых вычислительных процессов

Будем считать, что алгоритм задан последовательной программой линейного класса [3]. Основную вычислительную часть такой программы составляют циклы произвольной структуры вложенности; границы изменения параметров циклов задаются неоднородными формами, линейными по совокупности параметров циклов и внешних переменных; шаги изменения параметров циклов равны 1. Пусть в гнезде циклов имеется

Θ наборов выполняемых операторов. Под набором операторов обычно понимают один или несколько выполняемых операторов, окруженных одним и тем же множеством циклов. В этой работе, для простоты изложения и не ограничивая общности, будем считать, что оператор S_β составляет «набор» операторов с номером β. Выполняемые операторы линейно упорядочены расположением их в записи алгоритма.

Как уже отмечалось, тайлинг — это преобразование алгоритма для получения макроопераций, называемых также зерном вычислений или тайлами. При тайлинге каждый цикл разбивается на два цикла: глобальный, параметр которого определяет на данном уровне вложенности порядок вычисления тайлов, и локальный, в котором параметр исходного цикла изменяется в границах одного тайла. Если разбиение не производить и все итерации считать принадлежащими глобальному циклу, то получим так называемый глобальный не разбиваемый цикл; если все итерации отнести к локальному циклу, то получим локальный не разбиваемый цикл. Перестановкой и распределением циклов алгоритм преобразуется таким образом, чтобы глобальные циклы были внешними по отношению к локальным.

Размеры тайлов задаются натуральными числами $r_1^\beta,...,r_{n_\beta}^\beta$. Параметр r_ζ^β обозначает число значений параметра цикла j_ζ, приходящихся на один тайл β-го оператора. Если операторы S_α и S_β имеют общий цикл с параметром j_ζ, то $r_\zeta^\beta = r_\zeta^\beta$. Число частей, на которые при формировании тайлов разбивается область значений параметра j_ζ цикла, окружающего β-й оператор, обозначим Q_ζ^β Тайлы нумеруются по каждой координате в пределах от 0 до Q_ζ^β –1, $1 \le \zeta \le n_\beta$. Каждый тайл можно обозначить некоторым вектором J^{gl} или, подробнее, вектором $J^{gl}(j_1^{gl},...,j_{n_\beta}^{gl})$.

Рассмотрим следующий способ получения 2D зернистых (т.е. уровня макроопераций-тайлов) вычислительных процессов. Пусть оператор β исходного алгоритма окружен циклами с параметрами j_η и j_ξ. Произведем тайлинг и зафиксируем два глобальных цикла с уровнями вложенности η, ξ. К одному 2D процессу с координатами $(j_\eta^{gl}, j_\xi^{gl})$ отнесем вычисления всех тайлов с одинаковыми значениями функций (для каждого β — одна из функций).

$$\Pr^\beta\left(J^{gl}(j_1^{gl},...,j_{n_\beta}^{gl})\right) = (j_\eta^{gl}, j_\xi^{gl}), \tag{4}$$

$$\Pr^\beta\left(J^{gl}(j_1^{gl},...,j_{n_\beta}^{gl})\right) = (j_\eta^{gl}, Q_\xi^\beta - j_\xi^{gl} - 1), \tag{5}$$

$$\Pr^\beta\left(J^{gl}(j_1^{gl},...,j_{n_\beta}^{gl})\right) = (Q_\eta^\beta - j_\eta^{gl} - 1, j_\xi^{gl}). \tag{6}$$

Обозначим $(p_\eta,p_\xi) = (j_\eta^{gl}, j_\xi^{gl})$. Вычислительный процесс с координатами (p_η,p_ξ) будем обозначать $\Pr_{p_\eta,p_\xi}$.

Будем рассматривать распределение операций между 2D зернистыми вычислительными процессами как два независимых распределения операций между 1D процессами, т.е. будем независимо рассматривать первую и вторую координаты 2D процессов. При таком подходе для оценки коммуникационных затрат можно воспользоваться результатами для 1D процессов [11]. Утверждения, сформулированные в работе [11], позволяют получить асимптотические оценки объема коммуникационных операций для случая 2D процессов, задаваемых функциями вида (4)–(6). Для конкретных параллельных алгоритмов асимптотические оценки можно уточнять (заменить на точные выражения). За-

метим, что если есть «некоординатные» зависимости, то некоторая часть объема коммуникационных операций может учитываться дважды (в этой работе такая ситуация не возникает).

Параметр цикла j_ζ будет обозначать параметр j_η или j_ξ, т.е. под циклом уровня вложенности ζ понимается цикл уровня вложенности η или уровня вложенности ξ.

К одному 1D вычислительному процессу отнесем вычисления тайлов с одинаковыми значениями функций $\Pr^\beta\left(J^{gl}\right)$,$1\leq\beta\leq\Theta$, отображающих тайлы на процессы [5, 12, 13]. Будем полагать

$$\Pr^\beta\left(J^{gl}(j_1^{gl},...,j_{n_\beta}^{gl})\right)=j_\zeta^{gl} \tag{7}$$

или

$$\Pr^\beta\left(J^{gl}(j_1^{gl},...,j_{n_\beta}^{gl})\right)=Q_\zeta^\beta-j_\zeta^{gl}-1. \tag{8}$$

Функция $\Pr^\beta$ вида (7) или (8) задает вычислительный процесс $\Pr^\beta\left(J^{gl}\right)$ выполнения операций тайлов J^{gl} с одинаковыми значениями ζ-й координаты j_ζ^{gl} векторов J^{gl}.

Для возможности организовать полностью загруженные работой двумерные параллельные вычислительные процессы вида (4) достаточно, при определенных ограничениях на структуру и длину циклов, произвести тайлинг по трем координатам многомерного итерационного пространства [10]; для случая одномерной структуры вида (7) или (8) достаточно произвести тайлинг по двум координатам [12, 13].

Проиллюстрируем предлагаемый подход в следующем разделе. Рассматриваемый пример имеет и самостоятельное значение.

5. Распределение операций параллельного алгоритма, реализующего схему расщепления на 2D структуре процессов

Пусть $P^\eta\times P^\xi$ — число процессов, предназначенных для реализации алгоритма. Для простоты изложения будем считать числа $r_1=(N_1-1)/P^\eta$ и $r_2=(N_2-1)/P^\xi$ целыми.

Зафиксируем параметр j самого внешнего цикла. Пусть числа Q_3 и r_3 связаны соотношениями $r_3=\left\lceil\left(N_3-1\right)/Q_3\right\rceil$.

Зададим распределение операций между 2D зернистыми (т.е. уровня макроопераций-тайлов) вычислительными процессами. Будем считать, что по первой координате распределение операций, порождаемых операторами S_1, S_2, S_3, определяется циклами уровня вложенности 4, а распределение операций, порождаемых операторами S_4, S_5, S_6 и S_7, S_8, S_9, определяется параллельными циклами уровня вложенности 2; все эти циклы имеют параметр i_1, изменяющийся от 1 до N_1-1. Будем считать, что по второй координате распределение операций, порождаемых операторами S_1, S_2, S_3, определяется параллельным циклом уровня вложенности 2, распределение операций, порождаемых операторами S_4, S_5, S_6, определяется циклами уровня вложенности 4, а распределение операций, порождаемых операторами S_7, S_8, S_9, определяется параллельным циклом уровня вложенности 3; все эти циклы имеют параметр i_2, изменяющийся от 1 до N_2-1.

Опишем подробнее распределение операций между 2D зернистыми вычислительными процессами.

Для операций, порождаемых операторами S_1 и S_2, каждому процессу по первой координате назначим выполнить некоторое число r_1 итераций цикла с параметром i_1: первые r_1 итераций — процессу с номером (первой координаты) 0, следующие r_1 итераций — про-

цессу с номером 1, и так далее. Такой порядок выполнения итераций назовем прямым. Для операций, порождаемых операторами S_1 и S_2, каждому процессу по второй координате также назначим прямой порядок выполнения итераций: r_2 итераций цикла с параметром i_2. Операции этих итераций циклов с параметрами i_1 и i_2, а также r_3 итераций цикла с параметром i_3 составляют зерно вычислений, которое назовем тайлом $1a$ типа. Всего, при фиксированном j, одному процессу $\mathrm{Pr}_{p_\eta,p_\xi}$, $0 \le p_\eta \le P^\eta - 1$, $0 \le p_\xi \le P^\xi - 1$, назначается Q_3 тайлов $1a$ типа:

$$\mathrm{Pr}^{\beta}\left(J^{gl}(j,p_\xi,q_3,p_\eta)\right) = (p_\eta,p_\xi),\ \beta=1,\ \beta=2,\ q_3=0,1,\ldots,Q_3-1. \tag{9}$$

Для операций, порождаемых оператором S_3, каждому процессу по первой координате назначим так называемый обратный порядок выполнения r_1 итераций цикла с параметром i_1: первые r_1 итераций — процессу с номером $P^\eta - 1$, следующие r_1 итераций — процессу с номером $P^\eta - 2$, и так далее, процессу с номером 0 — последние r_1 итераций цикла. Для операций оператора S_3 по второй координате назначим, как и для операторов S_1 и S_2, прямой порядок выполнения итераций цикла с параметром i_2. Операции этих итераций циклов с параметрами i_1 и i_2, а также r_3 итераций цикла с параметром i_3 составляют зерно вычислений, которое назовем тайлом $1b$ типа. Всего, при фиксированном j, одному процессу $\mathrm{Pr}_{p_\eta,p_\xi}$, $0 \le p_\eta \le P^\eta - 1$, $0 \le p_\xi \le P^\xi - 1$, назначается Q_3 тайлов $1b$ типа:

$$\mathrm{Pr}^{\beta}\left(J^{gl}(j,p_\xi,q_3,p_\eta)\right) = (Q_\eta^\beta - p_\eta - 1,\ p_\xi),\ \beta=3,\ q_3=0,1,\ldots,Q_3-1. \tag{10}$$

Цикл с параметром i_1 является параллельным, но вместо этого естественного параллелизма по первой координате процессов используется конвейерный параллелизм. Это приводит к разгону и торможению вычислений, зато значительно улучшается локальность. Под разгоном (торможением) вычислений понимается число единиц времени, когда еще не все процессы начали (закончили) выполнение.

Для операций, порождаемых операторами S_4 и S_5, каждому процессу $\mathrm{Pr}_{p_\eta,p_\xi}$ как по первой, так и по второй координате назначим прямой порядок выполнения итераций (Q_3 тайлов $2a$ типа):

$$\mathrm{Pr}^{\beta}\left(J^{gl}(j,p_\eta,q_3,p_\xi)\right) = (p_\eta,p_\xi),\ \beta=4,\ \beta=5,\ \ q_3=0,1,\ldots,Q_3-1. \tag{11}$$

Для операций, порождаемых оператором S_6, каждому процессу назначим по первой координате прямой порядок выполнения итераций, а по второй координате назначим обратный порядок выполнения итераций (Q_3 тайлов $2b$ типа):

$$\mathrm{Pr}^{\beta}\left(J^{gl}(j,p_\eta,q_3,p_\xi)\right) = (p_\eta,Q_\xi^\beta - p_\xi - 1),\ \beta=6,\ q_3=0,1,\ldots,Q_3-1. \tag{12}$$

Параллельность цикла с параметром i_2 не используется (используется конвейерный параллелизм).

Для операций, порождаемых операторами S_7, S_8, S_9, каждому процессу $\mathrm{Pr}_{p_\eta,p_\xi}$ как по первой, так и по второй координате назначим прямой порядок выполнения итераций (тайл 3-го типа):

$$\mathrm{Pr}^{\beta}\left(J^{gl}(j,p_\eta,p_\xi,0)\right) = (p_\eta,p_\xi),\ \beta=7,\ \beta=8,\ \beta=9. \tag{13}$$

6. Исследование локальности параллельного алгоритма. Псевдокод алгоритма

Уровень вложенности, определяющий распределение операций $S_\beta(J)$ между процессами по первой координате, будем обозначать η^β. Уровень вложенности, определяющий распределение операций $S_\beta(J)$ между процессами по второй координате, будем обозначать ξ^β.

Имеем: η^β=4 для операторов S_β, $1\le\beta\le3$, η^β=2 для операторов S_β, $4\le\beta\le9$; ξ^β=2 для операторов S_β, $1\le\beta\le3$, ξ^β=4 для операторов S_β, $4\le\beta\le6$, ξ^β=3 для операторов S_β, $7\le\beta\le9$.

Утверждение 1. Пусть $N_1-1=N_2-1=N_3-1=M$. Если распределение операций между 2D-процессами задается формулами (9)–(13), то требуются коммуникационные операции только между процессами, номера которых отличаются по первой или по второй координате на 1. Суммарный объем коммуникационных операций, требуемых на одном слое, равен $5(P^\eta+P^\xi-2)M^2$.

Доказательство. Воспользуемся результатами работы [11] , при этом асимптотические оценки для рассматриваемого примера заменим на точные объемы данных.

Пусть вхождение (a,S_β,q) порождает истинную зависимость, задаваемую функцией $\overline{\mathbf{\Phi}}_{\alpha,\beta}(J)$. Обозначим через $\left(\mathbf{\Phi}_{\alpha,\beta}\right)_\zeta$ и $\left(\mathbf{\Psi}_{\alpha,\beta}\right)_\zeta$ строки матриц с номером ζ, а через $e^{(4)}_\zeta$ вектор-строку размера 4, у которой координата с номером ζ равна 1, а остальные координаты нулевые.

Приведем выкладки для второй координаты. Для первой координаты выкладки фактически приведены в работе [14].

Если $\left(\mathbf{\Phi}_{\alpha,\beta}\right)_{\zeta^\alpha}=e^{(4)}_{\zeta^\beta}$, $\left(\mathbf{\Psi}_{\alpha,\beta}\right)_{\zeta^\alpha}$=0, $\varphi^{\alpha,\beta}_{\zeta^\alpha}$=0, то коммуникационных операций не требуется. Данные условия выполняются для функций $\overline{\mathbf{\Phi}}_{1,1}$, $\overline{\mathbf{\Phi}}_{1,2}$, $\overline{\mathbf{\Phi}}_{2,2}$, $\overline{\mathbf{\Phi}}_{3,3}$, $\overline{\mathbf{\Phi}}_{1,3}$, $\overline{\mathbf{\Phi}}_{2,3}$, $\overline{\mathbf{\Phi}}_{7,7}$, $\overline{\mathbf{\Phi}}_{7,8}$, $\overline{\mathbf{\Phi}}_{8,8}$, $\overline{\mathbf{\Phi}}_{9,9}$, $\overline{\mathbf{\Phi}}_{7,9}$, $\overline{\mathbf{\Phi}}_{8,9}$, $\overline{\mathbf{\Phi}}_{9,2}$, поэтому эти функции коммуникационных операций не порождают.

Если $\left(\mathbf{\Phi}_{\alpha,\beta}\right)_{\zeta^\alpha}=e^{(4)}_{\zeta^\beta}$, $\left(\mathbf{\Psi}_{\alpha,\beta}\right)_{\zeta^\alpha}$=0, $\varphi^{\alpha,\beta}_{\zeta^\alpha}$=1 или $\varphi^{\alpha,\beta}_{\zeta^\alpha}$=–1, то коммуникационные операции происходят между процессами, номера которых отличаются на 1. Функции $\overline{\mathbf{\Phi}}_{4,4}$, $\overline{\mathbf{\Phi}}_{4,5}$, $\overline{\mathbf{\Phi}}_{5,5}$, $\overline{\mathbf{\Phi}}_{6,6}$ порождают коммуникационные операции суммарного объема $3(P^\xi-1)M^2$ (функции $\overline{\mathbf{\Phi}}_{4,4}$ и $\overline{\mathbf{\Phi}}_{4,5}$ порождают одну коммуникационную операцию). Функция $\overline{\mathbf{\Phi}}_{3,5}$ порождает коммуникационные операции объема $2(P^\xi-1)M^2$.

При выполнении операций оператора S_6 процессам назначается обратный порядок выполнения итераций. Поэтому, при α=6 или β=6 (но не $\alpha=\beta$=6) следует воспользоваться следующим результатом: если $\left(\mathbf{\Phi}_{\alpha,\beta}\right)_{\zeta^\alpha}=-e^{(4)}_{\zeta^\beta}$, $\left(\mathbf{\Psi}_{\alpha,\beta}\right)_{\zeta^\alpha}$=(0 0 1 0), $\varphi^{\alpha,\beta}_{\zeta^\alpha}$=0, то коммуникационных операций не требуется. Функции $\overline{\mathbf{\Phi}}_{4,6}$, $\overline{\mathbf{\Phi}}_{5,6}$, $\overline{\mathbf{\Phi}}_{6,8}$ коммуникационных операций не порождают.

Для первой координаты имеем [14]: функции $\overline{\mathbf{\Phi}}_{1,1}$, $\overline{\mathbf{\Phi}}_{1,2}$, $\overline{\mathbf{\Phi}}_{2,2}$, $\overline{\mathbf{\Phi}}_{3,3}$ порождают коммуникационные операции суммарного объема $3(P^\eta-1)M^2$ (функции $\overline{\mathbf{\Phi}}_{1,1}$ и $\overline{\mathbf{\Phi}}_{1,2}$ порождают одну коммуникационную операцию); функция $\overline{\mathbf{\Phi}}_{9,2}$ порождает коммуникационные операции объема $2(P^\eta-1)M^2$.

Суммарно процессы получают $5(P^{\eta}-1)M^2$ данных по первой координате и $5(P^{\xi}-1)M^2$ данных по второй координате. Итого — $5(P^{\eta}+P^{\xi}-2)M^2$ данных, коммуникационные операции происходят между процессами, номера которых отличаются только на 1 по первой или по второй координате.

Структуру кода выполнения алгоритма 2D процессами можно для каждого процесса $\mathrm{Pr}_{p_\eta,p_\xi}$, $0\le p_\eta\le P^{\eta}-1$, $0\le p_\xi\le P^{\xi}-1$, представить в следующем виде:

```
do j =1, j0
   do q3= 0, Q3-1
      sending and receiving caused by Φ9,2
      receiving caused by Φ1,1, Φ1,2 and caused by Φ2,2
      dopar i2= 1 + pξr2, min((pξ+1)r2, N2-1) // Tile 1a
         dopar i3= 1 + q3r3, min((q3+1)r3, N3-1)
            do i1= 1 + pηr1, (pη +1)r1
S1:            alpha(i2,i3,i1+1)=F1(alpha(i2,i3,i1))
S2:            beta(i2,i3,i1+1)=F2(alpha(i2,i3,i1),beta(i2,i3,i1),
                        y^j(i1,i2,i3),y^j(i1-1,i2,i3),y^j(i1+1,i2,i3))
      sending caused by Φ1,1, Φ1,2 and caused by Φ2,2
   do q3= 0, Q3 -1
      receiving caused by Φ3,3
      dopar i2= 1 + pξr2, min((pξ+1)r2, N2-1) // Tile 1b
         dopar i3= 1 + q3r3, min((q3+1)r3, N3-1)
            do i1= 1 + (Pη-pη-1)r1, (Pη-pη)r1
S3:            y^{j+1/3}(N1-i1,i2,i3)=
                        alpha(i2,i3,N1-i1+1)y^{j+1/3}(N1-i1+1,i2,i3)+
                        beta(i2,i3,N1-i1+1)
      sending caused by Φ3,3
   do q3= 0, Q3 -1
      sending and receiving caused by Φ3,5
      receiving caused by Φ4,4, Φ4,5 and caused by Φ5,5
      dopar i1= 1 + pηr1, (pη +1)r1 // Tile 2a
         dopar i3= 1 + q3r3, min((q3+1)r3, N3-1)
            do i2= 1 + pξr2, min((pξ+1)r2, N2-1)
S4:            alpha(i1,i3,i2+1)=F3(alpha(i1,i3,i2))
S5:            beta(i1,i3,i2+1)=F4(alpha(i1,i3,i2),beta(i1,i3,i2),
              y^{j+1/3}(i1,i2,i3),y^{j+1/3}(i1,i2-1,i3),y^{j+1/3}(i1,i2+1,i3))
      sending caused by Φ4,4, Φ4,5 and caused by Φ5,5
   dopar q3= 0, Q3 -1
      receiving caused by Φ6,6
      dopar i1= 1 + pη r1, (pη +1)r1 // Tile 2b
         dopar i3= 1 + q3r3, min((q3+1)r3, N3 -1)
            do i2= 1 + (Pξ-pξ-1)r2, min((Pξ-pξ)r2, N2 -1)
S6:            y^{j+2/3}(i1,N2-i2,i3)=
                        alpha(i1,i3,N2-i2+1) y^{j+2/3}(i1,N2-i2+1,i3)+
                        beta(i1,i3,N2-i2+1)
      sending caused by Φ6,6
   dopar i1= 1 + pηr1, (pη +1)r1 // Tile 3
      dopar i2= 1 + pξr2, min((pξ+1)r2, N2 -1)
         do i3= 1, N3-1
S7:         alpha(i3+1)=F5(alpha(i3))
S8:         beta(i3+1)=F6(alpha(i3),beta(i3),y^{j+2/3}(i1,i2,i3),
                        y^{j+2/3}(i1,i2,i3-1),y^{j+2/3}(i1,i2,i3+1))
         do i3= 1, N3-1
S9:         y^{j+1}(i1,i2,N3-i3)=
                  alpha(N3-i3+1) y^{j+1}(i1,i2,N3-i3+1)+beta(N3-i3+1)
```

Рис. 2. Структура MPI-кода (2D процессы), реализующего схему расщепления

Пусть $P^{\eta} \times P^{\xi} = P \times P = P^2$. Тогда объем коммуникационных операций, требуемых на одном слое, равен $10(P-1)M^2$. Если P^2 процессов организовать в одномерную структуру, то объем коммуникационных операций, требуемых на одном слое, равен $5(P^2-1)M^2$, что при большом числе процессов существенно больше $10(P-1)M^2$.

Кроме того, использование 2D структуры процессов позволяет, по сравнению со случаем 1D структуры (также использующей конвейерный параллелизм), существенно сократить разгон и торможение вычислений. Примем, что множество операций любого тайла выполняется за одну единицу времени. Тогда на каждом временном слое время разгона вычислений равно $2(P-1)$ при использовании 2D структуры и P^2-1 при использовании 1D структуры; такие же временные затраты и на торможение вычислений.

Отметим, что трехшаговая схема расщепления обладает естественным параллелизмом: при реализации схемы на параллельных компьютерах с распределенной памятью вычислительные процессы на каждом из трех шагов одного временного слоя могут выполняться независимо. Если P^2 процессов организовать в одномерную структуру, то объем коммуникационных операций, требуемых на одном слое, равен [14] $2\left(1-\frac{1}{P^2}\right)M^3$; кроме того, такая реализация требует обращение каждого процесса к локальной памяти всех других процессов. С ростом M и числа используемых процессов это приводит к большим накладным расходам на коммуникационные операции. Если P^2 процессов организовать в двумерную структуру (с использованием только естественного параллелизма), то не удается уменьшить объем коммуникационных операций.

7. Вычислительные эксперименты

Для вычислительного эксперимента был использован суперкомпьютер ЦОД МФТИ. Для сравнения эффективности двух реализаций: алгоритма с улучшенной локальностью, использующего 1D структуру процессов [14], и алгоритма, использующего 2D структуру процессов, будем использовать задачу (1)–(3) с заданным решением $u(x,t) = e^{3t+x_1+x_2+x_3}$.

Дополнительно в численном эксперименте алгоритмы сравниваются с реализацией алгоритма с естественным параллелизмом.

В табл. 1 и табл. 2 представлены результаты вычислительных экспериментов при $N_1=N_2=N_3=300$, $j_0=100$. Необходимо отметить, что время выполнения алгоритмов с улучшенной локальностью при фиксированном количестве процессов зависит от размера тайла $r_1 \times r_2 \times r_3$. Если размер тайла по координате отображения на процессы определяется автоматически, то размер тайла по свободным координатам можно определять при запуске алгоритма. За счет сокращения коммуникации между узлами суперкомпьютера проявляются преимущества разработанной версии алгоритма. В случае меньшей пересылки по сети (например, несколько процессов на одном узле) эффект менее заметен.

Для сравнения результатов при увеличении количества процессов были выполнены эксперименты с следующими параметрами: $N_1=N_2=N_3=400$ и $j_0=100$; количество процессов совпадает с количеством вычислительных узлов; параметры $r_2=r_3$ принимают значения 20, 24 и 28 (для 1D структуры); параметр r_3 принимает значения 20, 24 и 28 (для 2D структуры).

Таблица 1

Время работы алгоритма (N=300, 16 процессов на 16 узлах)

Алгоритм	r_2	r_3	**Время работы, с**
Естественный параллелизм	—	—	48.03
1D структура процессов	16	16	48.78
	20	20	47.50
	24	24	46.97
	28	28	46.61
	32	32	47.53
2D структура процессов	—	2	46.31
	—	6	44.19
	—	10	44.01
	—	18	44.07

Таблица 2

Время работы алгоритма (N=300, 16 процессов на 4 узлах)

Алгоритм	r_2	r_3	**Время работы, с**
Естественный параллелизм	—	—	42.62
1D структура процессов	16	16	36.40
	20	20	35.28
	24	24	36.26
	28	28	37.77
	32	32	43.88
2D структура процессов	—	2	37.10
	—	6	39.48
	—	10	40.57
	—	18	44.04

В табл. 3 и на рис. 3 представлены обобщенные результаты: для фиксированного числа процессов (узлов) выбрано наименьшее время работы алгоритма. Таким образом полученные данные показывают суммарный эффект применения тайлинга и предложенного подхода по уменьшению объема пересылаемых данных между узлами.

Экспериментальные результаты согласуются с оценкой эффективности разработанных алгоритмов в оптимизации пересылаемых данных, однако при увеличении количества используемых узлов полезный эффект становится менее заметным.

Приведенная параллельная версия алгоритма реализована на языке C++ с использованием MPI.

Таблица 3

Время работы алгоритма (N=400)

Алгоритм	**Время работы, с (количество процессов)**			
	4	**9**	**16**	**25**
Естественный параллелизм	170.97	111.02	48.70	33.28
1D структура процессов	135.97	82.01	50.07	36.00
2D структура процессов	135.11	78.74	46.31	32.64

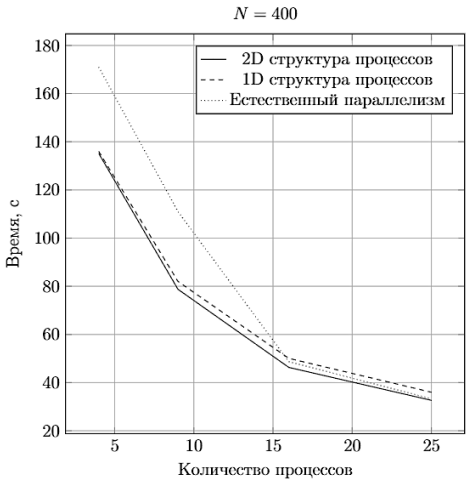


Рис. 3. Время выполнения алгоритмов на суперкомпьютере

Заключение

Время решения задачи на современном компьютере во многом определяется тем, насколько эффективно используется память с быстрым доступом. Особенно важное значение она имеет при организации параллельной обработки данных. Малая степень локализации данных оборачивается низкой эффективностью и масштабируемостью параллельных приложений [4]. В этой работе в качестве компьютера, на котором требуется реализовать алгоритм, рассмотрена многопроцессорная вычислительная система с распределенной памятью. Для этого класса компьютеров быстрой считается локальная память вычислительного узла. В работе исследована локальность нового параллельного алгоритма, реализующего схему расщепления численного решения трехмерного линейного уравнения теплопроводности. Алгоритм использует параллельные вычислительные процессы, логически организованные в двумерную структуру, обладает лучшей локальностью по сравнению с известными алгоритмами, использующими 1D структуры процессов.

На примере алгоритма, реализующего схему расщепления, представлен подход к организации параллельных 2D зернистых вычислительных процессов: распределение операций между процессами рассматривается как два независимых распределения операций между 1D процессами, анализируются информационные зависимостей, коммуникационные затраты оцениваются с помощью утверждений, доказанных для 1D процессов [11]. Отметим пригодность к автоматизации предлагаемого подхода: необходимая формальная информация об исходном алгоритме может быть получена с помощью свободно доступных библиотек и систем, проверка условий для получения объема и структуры коммуникационных операций 1D и 2D процессов допускает программную реализацию, конкретные значения заданных параметрически величин не требуются до компиляции.

Направления дальнейших исследований: автоматизация исследования локальности зернистых вычислительных процессов; разработка и программная реализация алгоритмов метода дробных шагов численного решения линейных и квазилинейных трехмерных параболических уравнений на суперкомпьютерах с распределенной памятью; разработка и программная реализация параллельных алгоритмов для решения прикладных задач с использованием предлагаемого подхода.

Работа выполнена в рамках государственной программы научных исследований Республики Беларусь «Конвергенция-2025», подпрограмма «Математические модели и методы».

Литература

1. Адуцкевич Е.В., Лиходед Н.А. Согласованное получение конвейерного параллелизма и распределения операций и данных между процессорами // Программирование. 2006. Т. 32, № 3. С. 54–65.
2. Баханович С.В., Лиходед Н.А., Мандрик П.А. Улучшение локальности параллельных алгоритмов численного решения двумерных квазилинейных параболических уравнений // Вестник Российского университета дружбы народов. Серия: Математика, информатика, физика. 2014. № 2. С. 211–215.
3. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. Санкт-Петербург: БХВ-Петербург, 2002. 608 с.
4. Воеводин Вл.В., Воеводин Вад.В. Спасительная локальность суперкомпьютеров // Открытые системы. 2013. № 9. С. 12–15.
5. Гергель В.П., Стронгин Р.Г. Основы параллельных вычислений для многопроцессорных вычислительных систем. Нижний Новгород: ННГУ им. Н.И. Лобачевского, 2003. 184 с.
6. Колешко С.Б., Попов Ф.Д. Механика жидкости и газа. Разностные схемы. Учеб. пособие. СПб.: Изд-во СПбГТУ, 2001. 72 с.
7. Корнеев Б.А., Левченко В. Д. Эффективное решение трехмерных задач газовой динамики Рунге—Кутты разрывным методом Галеркина// Журнал вычислительной математики и математической физики. 2016. № 3. С. 465–475. DOI: 10.7868/S0044466916030121.
8. Лиходед Н.А. Достаточные условия определения и использования данных в одном параллельном зернистом вычислительном процессе // Журнал вычислительной математики и математической физики. 2014. № 8. С. 122–133. DOI: 10.7868/s0044466914080092.
9. Лиходед Н.А., Баханович С.В., Жерело А.В. Получение аффинных преобразований для улучшения локальности гнезд циклов // Программирование. 2005. № 5. С. 52–65. DOI: 10.1007/s11086-005-0036-2.
10. Лиходед Н.А., Полещук М.А. Построение двумерных зернистых параллельных вычислительных процессов // Известия НАН Беларуси. Сер. физ.-мат. наук 2018. Т. 54, № 4. С. 417–426. DOI: 10.29235/1561-2430-2018-54-4-417-426.
11. Лиходед Н.А., Толстиков А.А. Метод оценки локальности параллельных алгоритмов, ориентированных на компьютеры с распределенной памятью // Доклады НАН Беларуси. 2020. Т. 64, № 6. С. 647–656. DOI: 10.29235/1561-8323-2020-64-6-647-656.
12. Лиходед Н.А., Толстиков А.А. Параллельные последовательности зернистых вычислений // Доклады НАН Беларуси. 2010. Т. 54, № 4. С. 36–41.
13. Толстиков А.А., Лиходед Н.А. Корректность разбиений алгоритмов при организации зернистых параллельных вычислительных процессов // Международный конгресс по информатике: информационные системы и технологии (CSIST'2011) (Минск, 31 октября – 3 ноября 2011 г.). Минск: Белорусский государственный университет, 2011. Т. 2. С. 122–126.
14. Толстиков А.А., Лиходед Н.А. Параллельный алгоритм метода расщепления для реализации на суперкомпьютерах с распределенной памятью // Международная научная конференция «Параллельные вычислительные технологии» (ПаВТ'2020) (Пермь, 31 марта – 2 апреля 2020 г.). Короткие статьи и описания плакатов. Челябинск: Издательский центр ЮУрГУ, 2020. С. 287–297.

15. Яненко Н.Н. Метод дробных шагов решения многомерных задач математической физики. Новосибирск: Изд-во «Наука». Сибирское отделение, 1967. 197 с.
16. Bondhugula U., Baskaran M., Krishnamoorthy S., Ramanujam J., Rountev A., Sadayappan P. Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model // Lecture Notes in Computer Science. Vol. 4959. Springer, 2008. P. 132–146. DOI: 10.1007/978-3-540-78791-4_9.
17. Buluc A., Gilberta J.R., Budak C. Solving path problems on the GPU // Parallel Computing. 2010. Vol. 36, no. 5–6. P. 241–253. DOI: 10.1016/j.parco.2009.12.002.
18. Dathathri R., Mullapudi R.T., Bondhugula U. Compiling affine loop nests for a dynamic scheduling runtime on shared and distributed memory // ACM Transactions on Parallel Computing (TOPC). 2016. Vol. 3, no. 2. DOI: 10.1145/2948975.
19. Lim A.W., Lam M.S. Maximizing parallelism and minimizing synchronization with affine partitions // Parallel Computing. 1998. Vol. 24, no. 3–4. P. 445–475. DOI: 10.1016/S0167-8191(98)00021-0.
20. Xue J. Loop tiling for parallelism. Springer Science & Business Media, 2000. 256 p. DOI: 10.1007/978-1-4615-4337-4.
21. Zhao J, Cohen A. Flextended tiles: a flexible extension of overlapped tiles for polyhedral compilation // ACM Transactions on Architecture and Code Optimization. 2019. Vol. 16, no. 4, article 47. DOI: 10.1145/3369382.
22. Zhao J, Di P. Optimizing the Memory Hierarchy by Compositing Automatic Transformations on Computations and Data. 53rd IEEE/ACM International Symposium on Microarchitecture (MICRO 2020) (Athens, Greece, October 17–21, 2020). IEEE, 2020. P. 427–441. DOI: 10.1109/micro50266.2020.00044.

Толстиков Алексей Александрович, старший преподаватель кафедры вычислительной математики факультета прикладной математики и информатики Белорусского государственного университета (Минск, Республика Беларусь)

Баханович Сергей Викторович, к.ф.-м.н., заместитель директора по научной и инновационной работе ГНУ «Институт математики НАН Беларуси» (Минск, Республика Беларусь)

Лиходед Николай Александрович, д.ф.-м.н., профессор кафедры вычислительной математики факультета прикладной математики и информатики Белорусского государственного университета (Минск, Республика Беларусь)

DOI: 10.14529/cmse210303

AN APPROACH TO ESTIMATE THE LOCALITY OF GRAINED COMPUTATION PROCESSES ORGANIZED TO A TWO-DIMENSIONAL STRUCTURE

© 2021 A.A. Tolstsikau[1], S.V. Bakhanovich[2], N.A. Likhoded[1]
[1]*Belarusian State University (Nezavisimosti Avenue 4, Minsk, 220030 Belarus),*
[2]*Institute of Mathematics NAS of Belarus (str. Surganova 11, Minsk, 220072 Belarus)*
E-mail: tolstikov@bsu.by, bsv@im.bas-net.by, likhoded@bsu.by
Received: 31.07.2021

Implementing algorithms for distributed memory computers, one should pay attention to locality, a computational property of the algorithm that reflects the usage of fast access memory, that plays an important role in achieving high performance. For computing systems with distributed memory the local memory of each node is considered fast. Usually the parallel algorithms have better locality if it has a smaller total size of communication data and a better structure of communication operations. In this work, a new parallel algorithm for the numerical solution of a three-dimensional linear heat equation is constructed on the basis of a splitting method with weights. The algorithm is specified for distributed memory computers, combines pipeline and natural parallelism, and uses a 2D process structure. For the case of 1D process structure it was shown that usage of a pipeline parallelism instead of part of natural parallelism reduces the total size of communication operations. The proposed 2D algorithm generalizes the known 1D algorithm. The usage of two-dimensional structures allows to reduce the total volume of communication operations and to reduce idle time of a pipeline. Therefore, the 2D algorithm has better locality compared to the 1D process structure. The computational experiments on a distributed computing sytem have shown the advantage of a new parallel algorithm. Proposed method can be used to obtain and to examine parallel algorithms for other schemes of the fractional step method. Additionally, the implemented splitting scheme algorithm shows an approach to obtain asymptotic estimation of the communication volume, presented method operates with granular (the macro-operation level) parallel computing processes organized logically into a two-dimensional structure. The estimations can be used to compare communication cost for alternative versions of parallel algorithms. The approach is developed on the basis of early result that allows to obtain asymptotic estimations of the communication operations volume of computational processes organized in a one-dimensional structure.

Keywords: parallel computing, distributed memory parallel computer, data communication reduction, fractional step method.

FOR CITATION

Tolstsikau A.A., Bakhanovich S.V., Likhoded N.A. An Approach to Extimate the Locality of Grained Computation Processes Organized to a Two-dimensional Structure. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2021. Vol. 10, no. 3. P. 37–55. (in Russian) DOI: 10.14529/cmse210303.

References

1. Adutskevich E.V., Likhoded N.A. A consistent generation of pipeline parallelism and distribution of operations and data among processors. Programming and Computer Software. 2006. Vol. 32, no. 3. P. 166–176.
2. Bakhanovich S.V., Likhoded N.A., Mandrik P.A. Improvement of locality of parallel algorithms of the numerical solutions of the two-dimensional quasilinear parabolic equation. RUDN Journal of Mathematics, Information Sciences and Physics. 2014. No. 2. P. 211–215. (in Russian)

3. Voevodin V.V., Voevodin Vl.V. Parallel Computing. St. Petersburg, BKhV-Petersburg Publ., 2002. 608 p. (in Russian)
4. Voevodin Vl.V., Voevodin Vad.V. The fortunate locality of supercomputers. Open Systems. 2013. No. 9. P. 12–15. (in Russian)
5. Gergel V.P., Strongin R.G. Introduction to parallel computing for multiprocessor systems. Nizhny Novgorod, Lobachevsky State University of Nizhny Novgorod, 2003. 184 p. (in Russian)
6. Koleshko S.B., Popov F.D. Mechanics of Liquids and Gases. Difference schemes. Tutorial. St. Petersburg, SPbGTU Publ., 2001. 72 p. (in Russian)
7. Korneev B.A., Levchenko V.D. Effective solving of three-dimensional gas dynamics problems with the Runge-Kutta discontinuous Galerkin method. Comput. Math. and Math. Phys. 2016. Vol. 56. P. 460–469. DOI: 10.1134/S0965542516030118.
8. Likhoded N.A. Sufficient conditions for the determination and use of data in the same granular parallel computation process. Computational Mathematics and Mathematical Physics. 2014. Vol. 54, no. 8. P. 1316–1326. (in Russian) DOI: 10.1134/s0965542514080077.
9. Likhoded N.A., Bakhanovich S.V., Zherelo A.V. Obtaining affine transformations to improve the locality of loop nests. Programming and Computer Software. 2005. Vol. 31, no. 5. P. 270–281. (in Russian) DOI: 10.1007/s11086-005-0036-2.
10. Likhoded N.A., Poleshchuk M.A. Tiled parallel 2D computational processes. Proceedings of the National Academy of Sciences of Belarus. Physics and Mathematics Series. 2018. Vol. 54, no. 4. P. 417–426. (in Russian) DOI: 10.29235/1561-2430-2018-54-4-417-426.
11. Likhoded N.A., Tolstsikau A.A. Locality estimation of parallel algorithm for distributed memory computers. Proceedings of the National Academy of Sciences of Belarus. 2020. Vol. 64, no. 6. P. 647–656. (in Russian) DOI: 10.29235/1561-8323-2020-64-6-647-656.
12. Likhoded N.A., Tolstsikau A.A. Parallel sequences of grain computations. Proceedings of the National Academy of Sciences of Belarus. 2010. Vol. 54, no. 4. P. 36–41. (in Russian)
13. Tolstsikau A.A., Likhoded N.A. Algorithm partition correctness while organizing grained parallel computing processes. International Congress on Computer Science: Information Systems and Technologies (CSIST'2011) (Minsk, Belarus, October 31 – November 3, 2011). Minsk, Belarusian State University, 2011. Vol. 2. P. 122–126. (in Russian)
14. Tolstsikau A.A., Likhoded N.A. Parallel algorithm implementing split method for a distributed memory computer. Parallel Computational Technologies (PCT 2020) (Perm, Russia, March 31 – April 2, 2020). Short papers and posters. Chelyabinsk, Publishing of the South Ural State University, 2020. P. 287–297. (in Russian)
15. Yanenko N.N. The method of fractional steps for solving multi-dimensional problems of mathematical physics. Novosibirsk, Science, 1967. 196 p. (in Russian)
16. Bondhugula U., Baskaran M., Krishnamoorthy S., Ramanujam J., Rountev A., Sadayappan P. Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model. Lecture Notes in Computer Science. Vol. 4959. Springer, 2008. P. 132–146. DOI: 10.1007/978-3-540-78791-4_9.
17. Buluc A., Gilberta J.R., Budak C. Solving path problems on the GPU. Parallel Computing. 2010. Vol. 36, no. 5–6. P. 241–253. DOI: 10.1016/j.parco.2009.12.002.
18. Dathathri R., Mullapudi R.T., Bondhugula U. Compiling affine loop nests for a dynamic scheduling runtime on shared and distributed memory. ACM Transactions on Parallel Computing (TOPC). 2016. Vol. 3, no. 2. DOI: 10.1145/2948975.

19. Lim A.W., Lam M.S. Maximizing parallelism and minimizing synchronization with affine partitions. Parallel Computing. 1998. Vol. 24, no. 3–4. P. 445–475. DOI: 10.1016/S0167-8191(98)00021-0.
20. Xue J. Loop tiling for parallelism. Springer Science & Business Media, 2000. 256 p. DOI: 10.1007/978-1-4615-4337-4.
21. Zhao J., Cohen A. Flextended tiles: a flexible extension of overlapped tiles for polyhedral compilation. ACM Transactions on Architecture and Code Optimization. 2019. Vol. 16, no. 4, article 47. DOI: 10.1145/3369382.
22. Zhao J., Di P. Optimizing the Memory Hierarchy by Compositing Automatic Transformations on Computations and Data. 53rd IEEE/ACM International Symposium on Microarchitecture (MICRO 2020) (Athens, Greece, October 17–21, 2020). IEEE, 2020. P. 427–441. DOI: 10.1109/micro50266.2020.00044.

УДК 004.021, 004.032.24, 004.051, 004.272 DOI: 10.14529/cmse210304

ПРИМЕНЕНИЕ КОНЦЕПЦИИ Q-ДЕТЕРМИНАНТА ДЛЯ ЭФФЕКТИВНОЙ РЕАЛИЗАЦИИ ЧИСЛЕННЫХ АЛГОРИТМОВ НА ПРИМЕРЕ МЕТОДА СОПРЯЖЕННЫХ ГРАДИЕНТОВ ДЛЯ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ

© 2021 В.Н. Алеева, М.Б. Шатов

Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76)
E-mail: aleevavn@susu.ru, charming.flurry@yandex.ru

Поступила в редакцию: 24.05.2021

Проблема повышения эффективности параллельных вычислений чрезвычайно актуальна. В статье продемонстрировано применение концепции Q-детерминанта для эффективной реализации численных алгоритмов на примере метода сопряженных градиентов для решения систем линейных уравнений. Концепция Q-детерминанта основана на унифицированном представлении численных алгоритмов в форме Q-детерминанта. Любой численный алгоритм имеет Q-детерминант. Q-детерминант состоит из Q-термов. Их число равно числу выходных данных алгоритма. Каждый Q-терм описывает все возможные способы вычисления одного из выходных данных на основе входных данных. Q-детерминант позволяет выразить и оценить внутренний параллелизм алгоритма, а также показать способ его параллельного исполнения. В работе приведены основные понятия концепции Q-детерминанта, необходимые для понимания приведенного исследования. Также описан основанный на концепции Q-детерминанта метод проектирования эффективных программ для численных алгоритмов. Результатом применения метода является программа, полностью использующая ресурс параллелизма алгоритма. Такая программа называется Q-эффективной. В качестве применения метода проектирования Q-эффективных программ описано проектирование программ для реализации метода сопряженных градиентов на параллельных вычислительных системах с общей и распределенной памятью. Приведены также результаты экспериментального исследования разработанных программ, проведенного с помощью суперкомпьютера «Торнадо ЮУрГУ».

Ключевые слова: повышение эффективности параллельных вычислений, Q-детерминант алгоритма, представление алгоритма в форме Q-детерминанта, Q-эффективная реализация алгоритма, ресурс параллелизма алгоритма, Q-эффективная программа.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Алеева В.Н., Шатов М.Б. Применение концепции Q-детерминанта для эффективной реализации численных алгоритмов на примере метода сопряженных градиентов для решения систем линейных уравнений // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 3. С. 56–71. DOI: 10.14529/cmse210304.

Введение

Основной целью параллельных вычислений является ускорение решения вычислительных задач. Для достижения этой цели используется распараллеливание алгоритмов, применяемых для решения задач. Чем больше реализация алгоритма использует ресурс параллелизма алгоритма, тем большее ускорение решения задачи она обеспечивает и тем она эффективнее. Самая эффективная реализация алгоритма использует ресурс параллелизма алгоритма полностью. Для ее выполнения требуется одновременно применять больше вычислителей (процессоров, ядер) параллельной вычислительной системы (ПВС), чем для выполнения любой другой реализации алгоритма, так как одновременно нужно выполнять

большее количество операций. Вычислительные мощности ПВС растут, что способствует эффективной реализации алгоритмов. Однако, на практике, как правило, параллельные программы выполняют не самые эффективные реализации алгоритмов, следовательно, не применяют вычислительные ресурсы ПВС настолько, насколько допускают алгоритмы, а это приводит к потере ускорения решения вычислительных задач. Таким образом, проблема эффективной реализации алгоритмов на ПВС является актуальной.

Целью исследования является демонстрация применения концепции Q-детерминанта для эффективной реализации численных алгоритмов на примере метода сопряженных градиентов для решения систем линейных уравнений. Для достижения цели решаются следующие задачи:

1) построение Q-детерминанта алгоритма, выполняющего метод сопряженных градиентов;
2) описание эффективной реализации алгоритма, выполняющего метод сопряженных градиентов;
3) разработка эффективных программ для метода сопряженных градиентов, предназначенных для ПВС с общей и распределенной памятью.

Статья относится к направлению исследований, представленному работами [1, 2, 8–12], и вносит вклад в развитие данного направления.

Статья организована следующим образом. Раздел 1 содержит обзор работ по теме исследования. В разделе 2 приведены некоторые основные понятия концепции Q-детерминанта, используемые в статье. В разделе 3 изложен метод проектирования Q-эффективных программ. В разделе 4 описано применение метода проектирования Q-эффективных программ для эффективной реализации алгоритма, выполняющего метод сопряженных градиентов. В разделе 5 представлены результаты экспериментального исследования Q-эффективных программ, реализующих метод сопряженных градиентов. которые были разработаны в последнее время, а также приведен обзор Q-эффективных программ, разработанных ранее. Заключение содержит краткое изложение полученных результатов, выводы об их применении и описание одного из перспективных направлений дальнейшего исследования.

1. Обзор работ по теме исследования

Приведем обзор работ по теме исследования ресурса параллелизма численных алгоритмов и его реализации. Таких работ, использующих универсальные подходы, очень мало.

Во-первых, отметим работы [3, 23], где есть очень важные и развитые исследования параллельной структуры алгоритмов и программ для их реализации на ПВС. Для исследования используются графы алгоритмов. Эти исследования адаптированы в открытой энциклопедии свойств алгоритмов AlgoWiki [5, 13]. При определении и реализации ресурса параллелизма алгоритмов используется индивидуальный подход к каждому алгоритму. Программное обеспечение для анализа ресурса параллелизма алгоритмов не рассматривается. Не рассматривается также единый для алгоритмов метод проектирования параллельных программ, использующих весь ресурс параллелизма алгоритмов.

Во-вторых, предлагаются подходы к разработке параллельных программ. Это привело к созданию различных языков параллельного программирования и инструментов. T-система [20] — одна из таких разработок. Она является инструментом для программирования, обеспечивающим автоматическое динамическое распараллеливание программ. Однако в работах по данному направлению исследований не показано, что параллельные програм-

мы, созданные с использованием Т-системы, полностью используют ресурс параллелизма алгоритмов. Синтез параллельных программ — это еще один подход к созданию параллельных программ. Он заключается в разработке новых параллельных алгоритмов с использованием базы знаний параллельных алгоритмов для решения более сложных задач. Технология фрагментированного программирования, язык ее реализации и система программирования LuNA разработаны на основе метода синтеза параллельных программ [7]. Такой подход не решает проблему исследования ресурса параллелизма алгоритмов и использования его в полной мере. Для снятия ограничений на ресурсы ПВС используется проектирование параллельных программ с помощью функционального языка программирования, который не зависит от архитектуры ПВС. Примером такого направления исследований может служить работа [15]. Однако здесь также нет обоснования, что созданные программы используют весь ресурс параллелизма реализуемых алгоритмов.

В-третьих, существует множество исследований, заключающихся в разработке параллельных программ для конкретных алгоритмов или для конкретной архитектуры ПВС. Например, к таким исследованиям относятся [17, 18, 21, 24]. Подобные исследования повышают эффективность реализации конкретных алгоритмов или реализации алгоритмов на ПВС определенной архитектуры, но они не обеспечивают общего универсального подхода.

Приведенный обзор показывает, что существующие подходы при решении проблемы исследования и использования ресурса параллелизма алгоритмов либо малоэффективны, либо неприменимы, либо не являются универсальными. Возможно, данный обзор не является полным, так как он основан на доступных авторам источниках. Вместе с тем, как мы отмечали ранее, ресурс параллелизма алгоритмов при реализации на ПВС используется не полностью. Этот факт дает основание сделать вывод, что либо приемлемое решение проблемы исследования и использования ресурса параллелизма алгоритмов разработано, но не достаточно широко известно, поэтому не применяется, либо его нет.

2. Некоторые основные понятия концепции Q-детерминанта

Рассмотрим алгоритмическую проблему $\bar{y} = F(N, B)$, где $N = \{n_1, \ldots, n_k\}$ — множество параметров размерности проблемы или N — пустое множество, B — множество входных данных, $\bar{y} = \{y_1, \ldots, y_m\}$ — множество выходных данных, при этом целое число m является либо константой, либо значением вычислимой функции параметров N при условии, что $N \neq \varnothing$. Здесь n_i $(i \in \{1, \ldots, k\})$ равно любому положительному целому числу. Если $N = \{n_1, \ldots, n_k\}$, то через $\bar{N} = \{\bar{n}_1, \ldots, \bar{n}_k\}$ обозначим набор из k положительных целых чисел, где $\bar{n}_i$ — некоторое заданное значение параметра n_i для каждого $i \in \{1, \ldots, k\}$. Через $\{\bar{N}\}$ обозначим множество всех возможных k-наборов $\bar{N}$. Пусть $\mathfrak{A}$ — алгоритм для решения алгоритмической проблемы, Q — набор операций, используемых алгоритмом $\mathfrak{A}$.

Определение 1. Определим выражение над B и Q, как терм в стандартном смысле математической логики [4].

Определение 2. Мы называем выражение цепочкой длины n, если оно является результатом применения некоторой ассоциативной операции из Q к n выражениям.

Определение 3. Если $N = \varnothing$, то любое выражение w над B и Q мы называем безусловным Q-термом. Пусть $N \neq \varnothing$ и V — множество всех выражений над B и Q. Тогда любое отображение $w : \{\bar{N}\} \to V \cup \varnothing$ также называется безусловным Q-термом.

Определение 4. Пусть $N = \varnothing$ и w — безусловный Q-терм. Предположим, что выражение w над B и Q имеет значение логического типа при любой интерпретации переменных B. Тогда безусловный Q-терм w называется безусловным логическим Q-термом. Пусть $N \neq \varnothing$ и w — безусловный Q-терм. Если выражение $w(\bar{N})$ для каждого $\bar{N} \in \{\bar{N}\}$ имеет значение логического типа при любой интерпретации переменных B, то безусловный Q-терм w называется безусловным логическим Q-термом.

Определение 5. Пусть $u_1, \ldots, u_l$ — безусловные логические Q-термы, $w_1, \ldots, w_l$ — безусловные Q-термы. Тогда множество l пар $(\widehat{u}, \widehat{w}) = \{(u_i, w_i)\}_{i \in \{1,\ldots,l\}}$ называется условным Q-термом длины l.

Определение 6. Пусть $(\widehat{u}, \widehat{w}) = \{(u_i, w_i)\}_{i=1,2,\ldots}$ — счетное множество пар безусловных Q-термов. Предположим, что $\{(u_i, w_i)\}_{i \in \{1,\ldots,l\}}$ — условный Q-терм для любого $l < \infty$. Тогда мы называем $(\widehat{u}, \widehat{w})$ условным бесконечным Q-термом.

Опишем нахождение значения безусловного Q-терма w при интерпретации переменных B. Если $N = \varnothing$, то нахождение значения выражения w означает нахождение значения безусловного Q-терма w при любой интерпретации переменных B. Если $N \neq \varnothing$ и $w(\bar{N}) \neq \varnothing$, то $w(\bar{N})$ является выражением над B и Q. Можно найти значение выражения $w(\bar{N})$ при любой интерпретации переменных B. Конечно, мы опускаем значение $w(\bar{N}) = \varnothing$. Следовательно, мы находим значение безусловного Q-терма w при любой интерпретации переменных B. Теперь опишем нахождение значения условного Q-терма $(\widehat{u}, \widehat{w})$ при интерпретации переменных B. Пусть $N = \varnothing$. Находим значения выражений u_i, w_i для $i \in \{i = 1, \ldots, l\}$. При нахождении значений мы можем найти пару u_{i_0}, w_{i_0} такую, что u_{i_0} имеет значение **true**. Следовательно, мы можем найти значение w_{i_0}. Тогда считаем, что $(\widehat{u}, \widehat{w})$ имеет значение w_{i_0}. В противном случае считаем, что значение $(\widehat{u}, \widehat{w})$ при интерпретации переменных B не определено. Пусть $N \neq \varnothing$ и $\bar{N} \in \{\bar{N}\}$. Находим выражения $u_i(\bar{N}), w_i(\bar{N})$ для $i \in \{i = 1, \ldots, l\}$. При нахождении значений мы можем найти пару $u_{i_0}(\bar{N}), w_{i_0}(\bar{N})$ такую, что $u_{i_0}(\bar{N})$ имеет значение **true**. Следовательно, мы можем найти значение $w_{i_0}(\bar{N})$. Тогда мы считаем, что $(\widehat{u}, \widehat{w})$ имеет значение $w_{i_0}(\bar{N})$. В противном случае считаем, что значение $(\widehat{u}, \widehat{w})$ для $\bar{N}$ и при такой интерпретации переменных B не определено. Аналогично можно определить значение условного бесконечного Q-терма.

Определение 7. Пусть $M = \{1, \ldots, m\}$. Предположим, что алгоритм $\mathfrak{A}$ состоит в нахождении для каждого $i \in M$ значения y_i путем вычисления значения Q-терма f_i. Тогда набор Q-термов $\{f_i \mid i \in M\}$ называется Q-детерминантом алгоритма $\mathfrak{A}$. Система уравнений $\{y_i = f_i \mid i \in M\}$ называется представлением алгоритма $\mathfrak{A}$ в форме Q-детерминанта.

Определение 8. Процесс вычисления Q-термов $\{f_i \mid i \in M\}$ алгоритма $\mathfrak{A}$ называется реализацией алгоритма $\mathfrak{A}$. Реализация алгоритма $\mathfrak{A}$ называется параллельной, если существуют операции, которые выполняются одновременно.

Определение 9. Реализация алгоритма $\mathfrak{A}$ называется Q-эффективной, если Q-термы $\{f_i \mid i \in M\}$ вычисляются одновременно, операции при их вычислении выполняются по мере готовности, при этом, если несколько операций цепочки готовы к выполнению, то они выполняются по схеме сдваивания.

Замечание 1. Определение Q-эффективной реализации показывает, что она полностью использует ресурс параллелизма алгоритма.

Ресурс параллелизма алгоритма характеризуют его высота и ширина. Эти понятия рассматриваются в работах [2, 9–11]. В [2, 11] описана программная Q-система, разработанная для автоматизированного исследования ресурса параллелизма алгоритмов, а также сравнения ресурсов параллелизма алгоритмов, решающих одну и ту же алгоритмическую проблему.

Определение 10. Реализация алгоритма $\mathfrak{A}$ называется выполнимой, если одновременно должно выполняться конечное (непустое) множество операций.

Замечание 2. Существуют алгоритмы, Q-эффективная реализация которых невыполнима. Пример такого алгоритма приведен в работе [2].

3. Метод проектирования Q-эффективных программ

Блок-схема численного алгоритма позволяет разработать последовательную программу, реализующую алгоритм. Аналогично, используя представление численного алгоритма в форме Q-детерминанта, можно разработать параллельную программу, реализующую представленный алгоритм. Эта идея лежит в основе метода проектирования Q-эффективных программ. Модель концепции Q-детерминанта, которую мы называем базовой, позволяет исследовать только машинно-независимые свойства алгоритмов. Поэтому базовая модель была расширена, чтобы учесть особенности реализации алгоритмов на реальных ПВС. Расширенная модель концепции Q-детерминанта получена путем добавления моделей параллельных вычислений: PRAM [19] для общей памяти и BSP [22] для распределенной памяти. Метод проектирования Q-эффективных программ использует расширенную модель концепции Q-детерминанта и состоит из этапов:

1) построение Q-детерминанта алгоритма;
2) описание Q-эффективной реализации алгоритма;
3) разработка параллельной программы для выполнимой Q-эффективной реализации алгоритма.

На первых двух этапах метода используется базовая модель концепции Q-детерминанта, а на третьем этапе — расширенная модель. Программа, полученная с помощью данного метода, была названа Q-эффективной, а процесс ее разработки Q-эффективным программированием. Так как Q-эффективная программа выполняет Q-эффективную реализацию алгоритма, то она полностью использует ресурс параллелизма алгоритма. Таким образом, Q-эффективная программа имеет самый высокий параллелизм среди программ, реализующих алгоритм.

Для разработки Q-эффективной программы для общей памяти достаточно описания Q-эффективной реализации алгоритма. При разработке Q-эффективной программы для распределенной памяти следует учитывать, что данные должны обладать свойством локальности, иначе могут возникнуть многочисленные промахи кэша при их считывании из памяти, что приведет к снижению быстродействия. Обеспечить локальность данных позволяет распределение вычислений между вычислительными узлами ПВС, основанное на описании Q-эффективной реализации алгоритма. В нашем исследовании для распределения вычислений между вычислительными узлами применяется принцип «master-slave» [16]. При этом мы используем один вычислительный узел «master» и несколько вычислительных узлов «slave». Узел «master» обозначается буквой M, а множество узлов «slave» — буквой S. В этом исследовании при разработке Q-эффективных программ используется язык про-

граммирования $C++$, технология OpenMP для ПВС с общей памятью, технологии MPI и OpenMP для ПВС с распределенной памятью.

Дальнейшим развитием модели концепции Q-детерминанта может быть добавление моделей параллельных вычислений для вычислительных систем с другими архитектурными особенностями. Также для создания Q-эффективных программ можно использовать различные языки программирования и технологии параллельного программирования. Таким образом, для одного численного алгоритма существует потенциально бесконечное множество Q-эффективных программ, каждая из которых создается и используется в рамках определенной вычислительной инфраструктуры. По-видимому, из всех Q-эффективных программ для данного алгоритма не существует лучшей по производительности, но каждая из этих программ является наиболее эффективной для той вычислительной инфраструктуры, для которой она создавалась.

Более подробно метод проектирования Q-эффективных программ описан в работе [8].

4. Применение метода проектирования Q-эффективных программ для эффективной реализации метода сопряженных градиентов

Метод сопряженных градиентов для решения систем линейных уравнений [14] широко применяется на практике, поэтому его исследование с целью эффективной реализации является актуальным. Опишем метод сопряженных градиентов. Предположим, нужно решить систему линейных уравнений $A\vec{x} = \vec{b}$, где $A = [a_{ij}]_{i,j=1,\dots,n}$ — симметричная положительно определенная матрица, $\vec{x} = (x_1, \dots, x_n)^T$ и $\vec{b} = (b_1, \dots, b_n)^T$. Пусть $\vec{x}^0 = (x_1^0, \dots, x_n^0)^T$ — начальное приближение решения системы. Процесс решения системы можно представить как минимизацию функционала $(A\vec{x}, \vec{x}) - 2(\vec{b}, \vec{x})$. Минимизируя данный функционал с использованием подпространств Крылова, получаем алгоритм $\mathfrak{B}$, реализующий метод сопряженных градиентов.

Итерационный процесс алгоритма $\mathfrak{B}$ можно записать так:

$$\vec{r}^0 = \vec{b} - A\vec{x}^0, \tag{1}$$

$$\vec{z}^0 = \vec{r}^0, \tag{2}$$

$$\vec{x}^k = \vec{x}^{k-1} + \alpha_k \vec{z}^{k-1}, \text{ где } k \in \{1, 2, \dots\}, \tag{3}$$

$$\alpha_k = (\vec{r}^{k-1}, \vec{r}^{k-1})/(A\vec{z}^{k-1}, \vec{z}^{k-1}), \tag{4}$$

$$\vec{r}^k = \vec{r}^{k-1} - \alpha_k A\vec{z}^{k-1}, \tag{5}$$

$$\beta_k = (\vec{r}^k, \vec{r}^k)/(\vec{r}^{k-1}, \vec{r}^{k-1}), \tag{6}$$

$$\vec{z}^k = \vec{r}^k + \beta_k \vec{z}^{k-1}. \tag{7}$$

В результате применения метода решение системы линейных уравнений может быть достигнуто на итерации n, если отсутствует погрешность вычислений. Если реализовывать метод на ПВС, то погрешность вычислений существует, поэтому для получения решения количество итераций будет больше n. В качестве критерия остановки итерационного процесса будем использовать выполнение условия

$$\|\vec{r}^k\|/\|\vec{b}\| < \epsilon, \tag{8}$$

где $\|\vec{r}^k\|/\|\vec{b}\|$ — относительная невязка на k-м шаге итерации, ϵ — точность вычислений.

Продемонстрируем применение метода проектирования Q-эффективных программ для эффективной реализации алгоритма $\mathfrak{B}$.

Этап 1. Q-детерминант алгоритма $\mathfrak{B}$ состоит из n условных бесконечных Q-термов, а представление алгоритма $\mathfrak{B}$ в форме Q-детерминанта имеет вид

$$x_i = \{(u^0, x_i^0), (u^1, x_i^1), \dots, (u^k, x_i^k), \dots\}, \text{ где } i \in \{1, \dots, n\},$$

$u^l = \|\vec{r}^l\|/\|\vec{b}\| < \epsilon$ для любого $l \in \{0, 1, \dots\}$.

Этап 2. Опишем Q-эффективную реализацию алгоритма $\mathfrak{B}$. В соответствии с определением Q-эффективной реализации все Q-термы, составляющие Q-детерминант, должны вычисляться одновременно, при этом их операции должны выполняться по мере готовности. Во-первых, нужно вычислить Q-терм u^0. Если значение u^0 **true**, то вычисление заканчивается, так как получено решение с заданной точностью, при этом n-кортеж $\{x_i = x_i^0\}_{i\in\{1,\dots,n\}}$ является решением. Если значение u^0 **false**, то должно продолжаться вычисление Q-термов u^1 и x_i^1 для всех $i \in \{1, \dots, n\}$ одновременно. Предположим, что вычисление продолжается на шаге итерации $k \geq 1$, при этом вычислены Q-термы u^k и x_i^k для всех $i \in \{1, \dots, n\}$ и значением u^k является **true**. Тогда вычисление должно быть завершено, так как получено решение с заданной точностью, при этом решением является n-кортеж $\{x_i = x_i^k\}_{i\in\{1,\dots,n\}}$. Если значение u^k **false**, то должно продолжаться вычисление Q-термов u^{k+1} и x_i^{k+1} для всех $i \in \{1, \dots, n\}$ одновременно.

Приведем описание процесса вычисления Q-термов u^l и x_i^l для всех $l \in \{0, 1, \dots\}$ и $i \in \{1, \dots, n\}$. Вычисление u^0 состоит в вычислении по формуле (1) одновременно с вычислением $\|\vec{b}\|$, а затем по формуле (8) при $k = 0$. Мы видим, что вычисление $\vec{x}^1$ должно состоять из вычислений по формулам (4) при $k = 1$, а затем (3) при $k = 1$ с учетом того, что $\vec{z}^0 = \vec{r}^0$. Вычисление u^1 состоит в последовательном вычислении по формулам (5) при $k = 1$, а затем (8) при $k = 1$. Для подготовки второй итерации должно быть выполнено вычисление по формуле (6) при $k = 1$, а затем (7) при $k = 1$. Вычисление $\vec{x}^l$ для всех $l \in \{2, 3, \dots\}$ должно состоять из вычислений по формулам (4) при $k = l$, (3) при $k = l$, применяемым последовательно. Вычисление u^l для всех $l \in \{2, 3, \dots\}$ состоит в вычислении по формулам (5) при $k = l$, а затем (8) при $k = l$. Для подготовки следующей $(l + 1)$-й итерации должно быть выполнено вычисление по формуле (6) при $k = l$, а затем (7) при $k = l$. При вычислении по формулам операции должны выполняться по мере готовности к выполнению. При этом операции цепочки должны выполняться по схеме сдваивания.

Q-эффективная реализация алгоритма $\mathfrak{B}$ выполнима, так как при реализации одновременно необходимо выполнять конечное число операций.

Этап 3. Описание Q-эффективной реализации алгоритма $\mathfrak{B}$, полученное на втором этапе метода, можно использовать для разработки Q-эффективной программы для общей памяти. Опишем процесс реализации алгоритма $\mathfrak{B}$ на ПВС с распределенной памятью, применяя принцип «master-slave».

Каждая компонента вектора $\vec{r}^0$ (см. (1)) вычисляется на отдельном узле S. Если количество узлов S меньше n, то каждый из узлов S должен выполнять вычисления для нескольких компонент вектора $\vec{r}^0$. Перед вычислением r_i^0, где $i \in \{1, \dots, n\}$ узел S получает от узла M компоненту b_i вектора $\vec{b}$, строку матрицы A с номером i и вектор $\vec{x}^0$. Одновременно с вычислением вектора $\vec{r}^0$ вычисляется значение $\|\vec{b}\|$ на узле M . Вычислен-

ный вектор $\vec{r}^0$ передается на узел M. На узле M выполняется вычисление Q-терма u^0. Если значение u^0 **true**, то вычисление заканчивается.

Для определения коэффициента α_1 (см. (4)) вычисление $A\vec{z}^0$, учитывая, что $\vec{z}^0 = \vec{r}^0$, выполняется на узлах S по аналогии с вычислением $A\vec{x}^0$. Для этого узел M передает на узлы S вектор $\vec{r}^0$. Одновременно с операцией $A\vec{z}^0$ выполняется вычисление скалярного произведения $(\vec{r}^0, \vec{r}^0)$ на узле M. Полученные на узлах S результаты, которые являются компонентами вектора $A\vec{z}^0$, передаются на узел M для вычисления скалярного произведения векторов $A\vec{z}^0$ и $\vec{z}^0$. Завершает вычисление α_1 операция деления на узле M. После этого выполняется вычисление приближения решения $\vec{x}^1$ (см. (3)) на узле M.

Одновременно с вычислением $\vec{x}^1$ на узле M на некотором одном узле S вычисляется $\vec{r}^1$ (см. (5)). Использовать несколько узлов S в этом случае нецелесообразно. Перед вычислением $\vec{r}^1$ узел M передает на узел S вектор $\vec{r}^0$, коэффициент α_1, вектор $A\vec{z}^0$ и результат скалярного произведения $(\vec{r}^0, \vec{r}^0)$. На этом же узле S вычисляется коэффициент β_1 (см. (6)). Для этого находится скалярное произведение $(\vec{r}^1, \vec{r}^1)$ и выполняется операция деления. После этого на узле S вычисляется также вектор $\vec{z}^1$ (см. (7)). Вычисленные на узле S значения $\vec{r}^1$, $(\vec{r}^1, \vec{r}^1)$, β_1 и $\vec{z}^1$ передаются на узел M. На узле M вычисляется Q-терм u^1. Если значение u^1 **true**, то вычисление заканчивается.

Для определения коэффициента α_2 (см. (4)) вычисление $A\vec{z}^1$ выполняется аналогично вычислению $A\vec{z}^0$. Для этого узел M передает на узлы S вектор $\vec{z}^1$. Полученные на узлах S компоненты вектора $A\vec{z}^1$ передаются на узел M для вычисления скалярного произведения векторов $A\vec{z}^1$ и $\vec{z}^1$. Вычислять скалярное произведения $(\vec{r}^1, \vec{r}^1)$ не нужно, так как оно было вычислено в процессе вычисления коэффициента β_1. Завершает вычисление α_2 операция деления на узле M. Вычисление $\vec{x}^2$ (см. (3)) выполняется на узле M.

Одновременно с вычислением $\vec{x}^2$ вычисляется $\vec{r}^2$ (см. (5)) на одном из узлов S. Перед вычислением $\vec{r}^2$ узел M передает на узел S вектор $\vec{r}^1$, коэффициент α_2, вектор $A\vec{z}^1$, результат скалярного произведения $(\vec{r}^1, \vec{r}^1)$ и вектор $\vec{z}^1$. Коэффициент β_2 (см. (6)) вычисляется на этом же узле S. Для этого выполняется скалярное произведение $(\vec{r}^2, \vec{r}^2)$ и операция деления. Затем на узле S вычисляется вектор $\vec{z}^2$ (см. (7)). Вычисленные на узле S значения $\vec{r}^2$, $(\vec{r}^2, \vec{r}^2)$, β_2 и $\vec{z}^2$ передаются на узел M. На узле M выполняется вычисление Q-терма u^2. Если значение u^2 **true**, то вычисление заканчивается.

Все последующие итерации алгоритма $\mathfrak{B}$ выполняются по аналогии со второй итерацией.

5. Разработка и экспериментальное исследование Q-эффективных программ

Для выполнения Q-эффективной реализации алгоритма $\mathfrak{B}$ были разработаны Q-эффективные программы для общей памяти и для распределенной памяти с применением принципа «master-slave», проектирование которых описано в разделе 4. Также была разработана Q-эффективная программа для распределенной памяти, не использующая принцип «master-slave», и две программы, которые не являются Q-эффективными, при этом первая из них предназначена для общей памяти, а вторая для распределенной памяти с применением принципа «master-slave». Программы, не являющиеся Q-эффективными, выполняют реализацию алгоритма $\mathfrak{B}$, отличающуюся от Q-эффективной реализации тем, что вычисления по формулам (5), (6) и (7) выполняются последовательно.

Для всех разработанных программ было проведено экспериментальное исследование динамических характеристик — времени выполнения и ускорения. Оно проводилось на суперкомпьютере «Торнадо» Южно-Уральского государственного университета. Для общей памяти был использован один вычислительный узел, для распределенной памяти — несколько вычислительных узлов. В экспериментах были задействованы два центральных процессора вычислительных узлов с многоядерными ускорителями Intel Xeon X5680 с частотой 3.33 GHz, каждый из которых имеет 6 ядер и поддерживает 12 потоков, оперативная память узла 24 Гб ECC DDR3 Full buffered [6]. При экспериментальном исследовании находилось решение системы линейных уравнений с плотной матрицей A, имеющей размер $n = 1000$, которая была сгенерирована с помощью специально разработанного программного обеспечения. Компоненты векторов $\vec{b}$ и $\vec{x}^0$ формировались путем генерации случайных чисел.

На рис. 1 показаны графики времени выполнения и ускорения программ для общей памяти. Рисунок 2 содержит графики времени выполнения и ускорения программ для распределенной памяти.

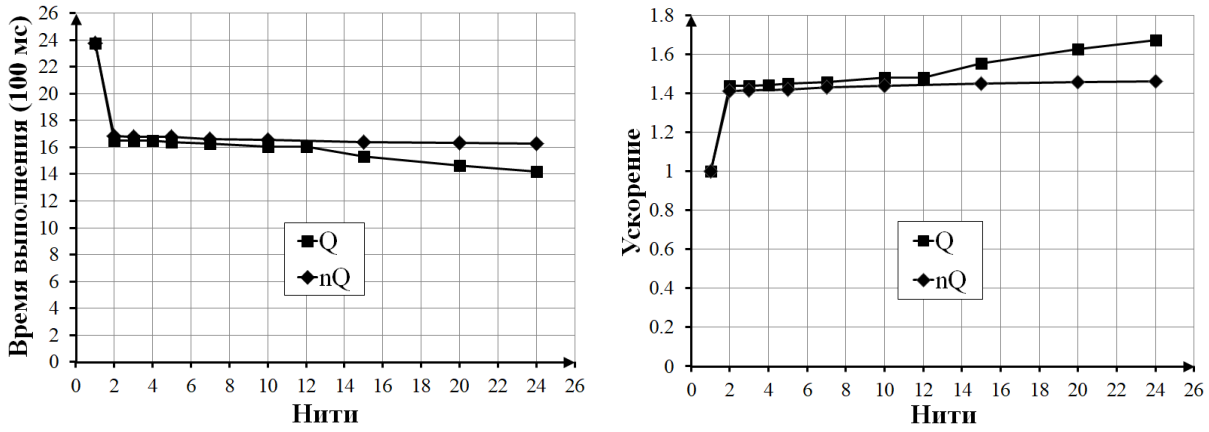


Рис. 1. Время выполнения и ускорение программ для общей памяти (Q — Q-эффективная программа, nQ — не Q-эффективная программа)

Поясним результаты экспериментального исследования.

На динамические характеристики Q-эффективной программы влияет множество факторов. Однако главное влияние оказывает ресурс параллелизма алгоритма, который программа выполняет. Ресурсы параллелизма у алгоритмов разные. Существуют алгоритмы, все операции которых могут быть выполнены параллельно. Примером такого алгоритма является алгоритм сложения двух векторов. Но есть также алгоритмы, которые не имеют параллельных реализаций, например, алгоритм, Q-детерминант которого состоит из одного безусловного Q-терма, имеющего вид

$$(\dots((((a_1 + a_2) \times a_3 + a_4) \times a_5 + a_6) \times a_7 + a_8)\dots) \times a_{2n-1} + a_{2n},$$

где n — параметр размерности. Подавляющее большинство алгоритмов имеют ресурс параллелизма, находящийся между этими двумя крайностями. Любой подход к проектированию параллельных программ основывается на распараллеливании выполняемых программой алгоритмов в рамках их ресурса параллелизма. Следовательно, значения динамических характеристик проектируемых программ из-за разных ресурсов параллелизма выполняемых алгоритмов будут разными.

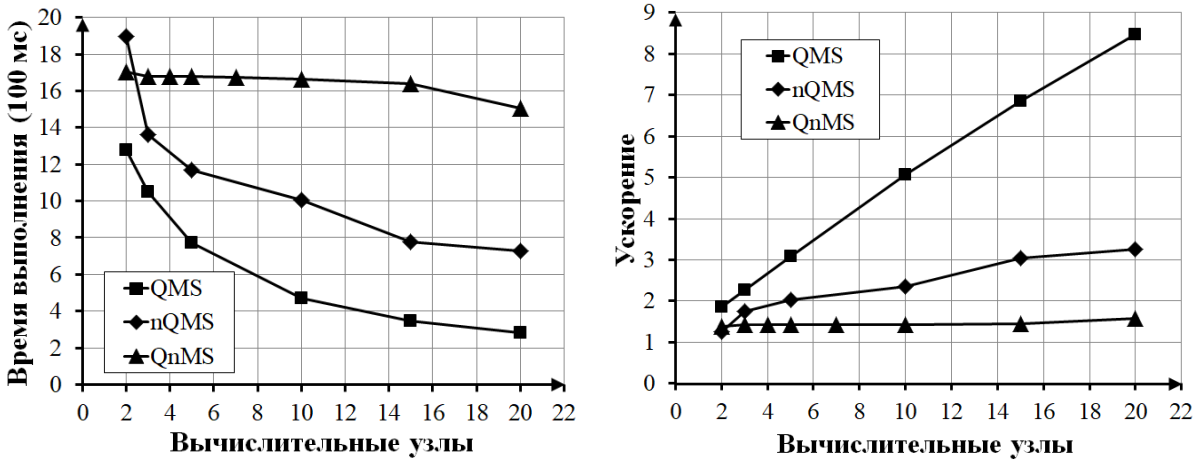


Рис. 2. Времея выполнения и ускорение программ для распределенной памяти (QMS — Q-эффективная программа, использующая принцип «master-slave», nQMS — не Q-эффективная программа, использующая принцип «master-slave», QnMS — Q-эффективная программа, не использующая принцип «master-slave»)

Проводя экспериментальное исследование Q-эффективных программ, можно лишь констатировать, каковы значения их динамических характеристик, но улучшить эти значения, не меняя используемую вычислительную инфраструктуру, то есть условия разработки и исполнения программ, не удастся. С помощью данного исследования получены значения динамических характеристик Q-эффективных программ, выполняющих алгоритм $\mathfrak{B}$, который реализует метод сопряженных градиентов. Если эти значения не устраивают пользователя, то для решаемой алгоритмической проблемы возможно с помощью программной Q-системы [2, 11] подобрать алгоритм с лучшим ресурсом параллелизма, чем алгоритм $\mathfrak{B}$.

Кроме ресурса параллелизма выполняемого параллельной программой алгоритма отметим и другие факторы, которые могут существенно влиять на динамические характеристики программы.

Следует подчеркнуть, что, как известно, время выполнения параллельных программ увеличивают пересылки информации между вычислительными узлами ПВС, а также синхронизация вычислений.

Если при реализации на ПВС Q-эффективной программе выделено не достаточное количество вычислительных ресурсов, в нашем случае вычислительных ядер, узлов, чтобы могла быть выполнена Q-эффективная реализация алгоритма, то значения динамических характеристик ухудшаются. Разработанные в данном исследовании Q-эффективные программы используют ресурс параллелизма алгоритма $\mathfrak{B}$ полностью, но выделенных ресурсов ПВС не достаточно для выполнения Q-эффективной реализации, поэтому полученные значения динамических характеристик всех трех Q-эффективных программ Q (рис. 1), QMS и QnMS (рис. 2) хуже, чем могли бы быть, если бы ресурсов было достаточно. Особенно нехватка ресурсов ПВС повлияла на время выполнения и ускорение Q-эффективной программы Q для общей памяти. Отметим, что на динамические характеристики программ nQ (рис. 1) и nQMS (рис. 2), не являющихся Q-эффективными, нехватка ресурсов ПВС также оказала влияние.

На рис. 2 можно видеть, что Q-эффективная программа QMS для распределенной памяти с применением принципа «master-slave» демонстрирует наивысшее среди соперников ускорение. Объясним этот факт.

Лучшие динамические характеристики программы QMS по сравнению с программой QnMS, по нашему мнению, можно объяснить, в частности, тем, что при разработке программы QMS обеспечивается свойство локальности данных за счет распределения вычислений по узлам, а при разработке программы QnMS нет, в результате чего возникают многочисленные промахи кэша при считывании данных из памяти. Обеспечение локальности данных при параллельных вычислениях имеет важное значение, так как повышает быстродействие программ.

Если программа, не являющаяся Q-эффективной, и Q-эффективная программа имеют одинаковые условия разработки и исполнения, то динамические характеристики программы, не являющейся Q-эффективной, хуже, чем Q-эффективной, так как она использует не весь ресурс параллелизма алгоритма. Результаты экспериментального исследования программ QMS и nQMS (рис. 2), а также программ Q и nQ (рис. 1) подтверждают это.

Все три Q-эффективные программы Q (рис. 1), QMS и QnMS (рис. 2) соответствуют разным вычислительным инфраструктурам. С помощью экспериментального исследования мы установили, что эти Q-эффективные программы имеют разное быстродействие. Но даже при влиянии на динамические характеристики перечисленных выше факторов каждая из Q-эффективных программ является наиболее эффективной для своей вычислительной инфраструктуры.

Применение метода проектирования параллельных программ для Q-эффективной реализации алгоритмов с Q-детерминантами различной структуры рассмотрено ранее в ряде выпускных квалификационных работ студентов Южно-Уральского государственного университета. Их обзор приведен в [9]. В этих работах были исследованы алгоритмы умножения плотных и разреженных матриц, метод Гаусса—Жордана, метод Якоби для решения систем линейных уравнений. Рассмотрен также метод прогонки для решения систем трехточечных уравнений, имеющий малый ресурс параллелизма. В этом случае не следует использовать распределенную память, поскольку это может привести к снижению производительности. Также рассмотрен метод Фурье для решения системы разностных уравнений, обладающий большим ресурсом параллелизма. Его можно эффективно реализовать на распределенной памяти, используя принцип «master-slave». Следует отметить, что исследования, приведенные в работах, показали, что для некоторых алгоритмов использовать принцип «master-slave» не целесообразно, так как это может привести к увеличению обменов между вычислительными узлами. Примером в этом случае является алгоритм для реализации метода Якоби для решения систем пятиточечных разностных уравнений. Исследования показали также, что некоторые алгоритмы имеют сложные, но хорошо структурированные Q-детерминанты. Например, такие алгоритмы реализуют метод Гаусса—Жордана и метод Гаусса—Зейделя для решения систем линейных уравнений. Аналогичными свойствами обладают и многие другие известные алгоритмы. Хорошо структурированные Q-детерминанты алгоритмов упрощают создание Q-эффективных программ для этих алгоритмов.

Заключение

В статье показано применение метода проектирования Q-эффективных программ, основанного на концепции Q-детерминанта, для эффективной реализации численных алгорит-

мов на примере метода сопряженных градиентов для решения систем линейных уравнений. Для этого были решены следующие задачи:

1) построение Q-детерминанта алгоритма, выполняющего метод сопряженных градиентов;
2) описание Q-эффективной реализации алгоритма, выполняющего метод сопряженных градиентов;
3) разработка Q-эффективных программ для метода сопряженных градиентов, предназначенных для ПВС с общей и распределенной памятью, и их экспериментальное исследование.

Данное исследование пополнило коллекцию алгоритмов, для которых разработаны Q-эффективные программы и оценены их динамические характеристики.

Применение метода проектирования Q-эффективных программ решает проблему наиболее полного использования ресурса параллелизма численных алгоритмов и тем самым делает возможной эффективную реализацию численных алгоритмов на ПВС. Программная Q-система для исследования ресурса параллелизма численных алгоритмов [2, 11], метод проектирования Q-эффективных программ и технология Q-эффективного программирования [9] в комплексе являются одним из решений проблемы повышения эффективности параллельных вычислений, использующих численные алгоритмы. Их могут применять разработчики программного обеспечения для проектирования эффективных программ, предназначенных для любых ПВС — от персональных компьютеров с многоядерными процессорами до суперкомпьютеров. Это приведет к уменьшению времени выполнения программного обеспечения и повышению быстродействия вычислительных систем.

Одним из перспективных направлений развития описанных в статье исследований является автоматизированное проектирование и исполнение Q-эффективных программ. Исследования по данному направлению уже ведутся.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-00865 а и при поддержке Правительства РФ в соответствии с Постановлением № 211 от 16.03.2013 г. (соглашение № 02.A03.21.0011).

Литература

1. Алеева В.Н. Анализ параллельных численных алгоритмов. Препринт № 590. Новосибирск: ВЦ СО АН СССР, 1985. 23 с.

2. Алеева В.Н., Зотова П.С., Склезнев Д.С. Расширение возможностей исследования ресурса параллелизма численных алгоритмов с помощью программной Q-системы // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 2. С. 66–81. DOI: 10.14529/cmse210205.

3. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. СПб.: БХВ-Петербург, 2002. 608 с.

4. Ершов Ю.Л., Палютин Е.А. Математическая логика. М.: Наука, 1987. 336 с.

5. Открытая энциклопедия свойств алгоритмов. URL: `https://algowiki-project.org/ru` (дата обращения: 16.05.2021).

6. Суперкомпьютер «Торнадо ЮУрГУ». URL: `http://supercomputer.susu.ru/computers/tornado/` (дата обращения: 16.05.2021).

7. Akhmed-Zaki D., Lebedev D., Malyshkin V., et al. Automated Construction of High Performance Distributed Programs in LuNA System // Parallel Computing Technologies (PaCT 2019). Lecture Notes in Computer Science. Vol. 11657. 2019. P. 3–9. DOI: 10.1007/978-3-030-25636-4_1.

8. Aleeva V. Designing a Parallel Programs on the Base of the Conception of Q-Determinant // Supercomputing. RuSCDays 2018. Communications in Computer and Information Science. Vol. 965. 2019. P. 565–577. DOI: 10.1007/978-3-030-05807-4_48.

9. Aleeva V.N. Improving Parallel Computing Efficiency // Proceedings – 2020 Global Smart Industry Conference, GloSIC 2020. IEEE, 2020. P. 113–120. Article number 9267828. DOI: 10.1109/GloSIC50886.2020.9267828.

10. Aleeva V.N., Aleev R.Zh. High-Performance Computing Using Application of Q-determinant of Numerical Algorithms // Proceedings – 2018 Global Smart Industry Conference, GloSIC 2018. IEEE, 2018. 8 p. Article number 8570160. DOI: 10.1109/GloSIC.2018.8570160.

11. Aleeva V., Bogatyreva E., Skleznev A., et al. Software Q-system for the Research of the Resource of Numerical Algorithms Parallelism // Supercomputing. RuSCDays 2019. Communications in Computer and Information Science. Vol. 1129. 2019. P. 641–652. DOI: 10.1007/978-3-030-36592-9_52.

12. Aleeva V.N., Sharabura I.S., Suleymanov D.E. Software System for Maximal Parallelization of Algorithms on the Base of the Conception of Q-determinant // Parallel Computing Technologies (PaCT 2015). Lecture Notes in Computer Science. Vol. 9251. 2015. P. 3–9. DOI: 10.1007/978-3-319-21909-7_1.

13. Antonov A.S., Dongarra J., Voevodin V.V. AlgoWiki Project as an Extension of the Top500 Methodology // Supercomputing Frontiers and Innovations. 2018. Vol. 5, no. 1. P. 4–10. DOI: 10.14529/jsfi180101.

14. Henk A. van der Vorst. Iterative Krylov Methods for Large Linear System. USA: Cambridge University Press, 2003. 221 p. DOI: 10.1017/CBO9780511615115.

15. Legalov A.I., Vasilyev V.S., Matkovskii I.V., et al. A Toolkit for the Development of Data-Driven Functional Parallel Programmes // Parallel Computational Technologies (PCT'2018). Communications in Computer and Information Science. Vol. 910. 2018. P. 16–30. DOI: 10.1007/978-3-319-99673-8_2.

16. Leung J.Y.-T., Zhao H. Scheduling problems in master-slave model // Annals of Operations Research. 2008. Vol. 159. P. 215–231. DOI: 10.1007/s10479-007-0271-4.

17. Li Y., Dou W., Yang K., et al. Optimized Data I/O Strategy of the Algorithm of Parallel Digital Terrain Analysis // 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science. 2014. P. 34–37. DOI: 10.1109/DCABES.2014.10.

18. Matveev S.A., Zagidullin R.R., Smirnov A.P., et al. Parallel Numerical Algorithm for Solving Advection Equation for Coagulating Particles // Supercomputing Frontiers and Innovations. 2018. Vol. 5, no. 2. P. 43–54. DOI: 10.14529/jsfi180204.

19. McColl W.F. General Purpose Parallel Computing // Lectures on Parallel Computation, Cambridge International Series on Parallel Computation. USA: Cambridge University Press, 1993. P. 337–391.

20. Moskovsky A., Roganov V., Abramov S. Parallelism Granules Aggregation with the T-System // Parallel Computing Technologies (PaCT 2007). Lecture Notes in Computer Science. Vol. 4671. 2007. P. 293–302. DOI: 10.1007/978-3-540-73940-1_30.

21. Prifti V., Bala R., Tafa I., et al. The time profit obtained by parallelization of quicksort algorithm used for numerical sorting // Science and Information Conference (SAI). 2015. P. 897–901. DOI: 10.1109/SAI.2015.7237248.

22. Valiant L.G. A bridging model for parallel computation // Communications of the ACM. 1990. Vol. 33, no. 8. P. 103–111. DOI: 10.1145/79173.79181.

23. Voevodin V.V., Voevodin Vl.V. The V-Ray technology of optimizing programs to parallel computers // Numerical Analysis and Its Applications (WNAA 1996). Lecture Notes in Computer Science. Vol. 1196. 1997. P. 546–556. DOI: 10.1007/3-540-62598-4_136.

24. You J., Kezhang H., Liang H., et al. Research on parallel algorithms for calculating static characteristics of electromagnetic relay // IEEE 11th Conference on Industrial Electronics and Applications (ICIEA). 2016. P. 1421–1425. DOI: 10.1109/ICIEA.2016.7603808.

Алеева Валентина Николаевна, к.ф.-м.н., доцент, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Шатов Михаил Борисович, студент, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

DOI: 10.14529/cmse210304

APPLICATION OF THE Q-DETERMINANT CONCEPT FOR EFFICIENT IMPLEMENTATION OF NUMERICAL ALGORITHMS BY THE EXAMPLE OF THE CONJUGATE GRADIENT METHOD FOR SOLVING SYSTEMS OF LINEAR EQUATIONS

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia)
E-mail: aleevavn@susu.ru, charming.flurry@yandex.ru

Received: 24.05.2021

The problem of improving the efficiency of parallel computing is very topical. The article demonstrates the application of the concept of Q-determinant for the effective implementation of numerical algorithms by the example of the conjugate gradient method for solving systems of linear equations. The concept of the Q-determinant is based on a unified representation of numerical algorithms in the form of the Q-determinant. Any numerical algorithm has a Q-determinant. The Q-determinant consists of Q-terms. Their number is equal to the number of output data items. Each Q-term describes all possible ways to compute one of the output data items based on the input data. The Q-determinant allows you to express and evaluate the internal parallelism of the algorithm, as well as to show the method of its parallel execution. The article gives the main notions of the Q-determinant concept necessary for better understanding of our research. Also, we describe a method of designing effective programs for numerical algorithms on the base of the concept of the Q-determinant. As a result, we obtain the program which uses the parallelism resource of the algorithm completely, and this program is called Q-effective. As application of the method for design of Q-effective programs, we describe the designing programs for conjugate gradient method for implementation on parallel computing systems with shared and distributed memory. Finally, for developed programs we present the results of experiments on a supercomputer "Tornado SUSU".

Keywords: improving parallel computing efficiency, Q-determinant of algorithm, representation of algorithm in the form of Q-determinant, Q-effective implementation of algorithm, parallelism resource of algorithm, Q-effective program.

FOR CITATION

Aleeva V.N., Shatov M.B. Application of the Q-determinant Concept for Efficient Implementation of Numerical Algorithms by the Example of the Conjugate Gradient Method for Solving Systems of Linear Equations. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2021. Vol. 10, no. 3. P. 56–71. (in Russian) DOI: 10.14529/cmse210304.

References

1. Aleeva V.N. Analysis of Parallel Numerical Algorithms. Preprint no. 590. Novosibirsk, Computing Center of the Siberian Branch of the Academy of Sciences of the USSR, 1985. 23 p. (in Russian)
2. Aleeva V.N., Zotova P.S., Skleznev D.S. Advancement of research for the parallelism resource of numerical algorithms with help of software Q-system. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2021. Vol. 10, no. 2. P. 66–81. (in Russian) DOI: 10.14529/cmse210205.
3. Voevodin V.V., Voevodin Vl.V. Parallel Computing. St. Petersburg, BHV-Petersburg, 2002. 608 p. (in Russian)
4. Ershov Yu.L., Palyutin E.A. Mathematical Logic. Moscow, Mir, 1984. 303 p.
5. Open Encyclopedia of Parallel Algorithmic Features. URL: `https://algowiki-project.org/en` (accessed: 16.05.2021).
6. "Tornado SUSU" Supercomputer. URL: `http://supercomputer.susu.ru/en/computers/tornado/` (accessed: 16.05.2021).
7. Akhmed-Zaki D., Lebedev D., Malyshkin V., et al. Automated Construction of High Performance Distributed Programs in LuNA System. Parallel Computing Technologies (PaCT 2019). Lecture Notes in Computer Science. Vol. 11657. 2019. P. 3–9. DOI: 10.1007/978-3-030-25636-4_1.
8. Aleeva V. Designing a Parallel Programs on the Base of the Conception of Q-Determinant. Supercomputing. RuSCDays 2018. Communications in Computer and Information Science. Vol. 965. 2019. P. 565–577. DOI: 10.1007/978-3-030-05807-4_48.
9. Aleeva V.N. Improving Parallel Computing Efficiency. Proceedings – 2020 Global Smart Industry Conference, GloSIC 2020. IEEE, 2020. P. 113–120. Article number 9267828. DOI: 10.1109/GloSIC50886.2020.9267828.
10. Aleeva V.N., Aleev R.Zh. High-Performance Computing Using Application of Q-determinant of Numerical Algorithms. Proceedings – 2018 Global Smart Industry Conference, GloSIC 2018. IEEE, 2018. 8 p. Article number 8570160. DOI: 10.1109/GloSIC.2018.8570160.
11. Aleeva V., Bogatyreva E., Skleznev A., et al. Software Q-system for the Research of the Resource of Numerical Algorithms Parallelism. Supercomputing. RuSCDays 2019. Communications in Computer and Information Science. Vol. 1129. 2019. P. 641–652. DOI: 10.1007/978-3-030-36592-9_52.

12. Aleeva V.N., Sharabura I.S., Suleymanov D.E. Software System for Maximal Parallelization of Algorithms on the Base of the Conception of Q-determinant. Parallel Computing Technologies (PaCT 2015). Lecture Notes in Computer Science. Vol. 9251. 2015. P. 3–9. DOI: 10.1007/978-3-319-21909-7_1.

13. Antonov A.S., Dongarra J., Voevodin V.V. AlgoWiki Project as an Extension of the Top500 Methodology. Supercomputing Frontiers and Innovations. 2018. Vol. 5, no. 1. P. 4–10. DOI: 10.14529/jsfi180101.

14. Henk A. van der Vorst. Iterative Krylov Methods for Large Linear System. USA, Cambridge University Press, 2003. 221 p. DOI: 10.1017/CBO9780511615115.

15. Legalov A.I., Vasilyev V.S., Matkovskii I.V., et al. A Toolkit for the Development of Data-Driven Functional Parallel Programmes. Parallel Computational Technologies (PCT'2018). Communications in Computer and Information Science. Vol. 910. 2018. P. 16–30. DOI: 10.1007/978-3-319-99673-8_2.

16. Leung J.Y.-T., Zhao H. Scheduling problems in master-slave mode. Annals of Operations Research. 2008. Vol. 159. P. 215–231. DOI: 10.1007/s10479-007-0271-4.

17. Li Y., Dou W., Yang K., et al. Optimized Data I/O Strategy of the Algorithm of Parallel Digital Terrain Analysis. 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science. 2014. P. 34–37. DOI: 10.1109/DCABES.2014.10.

18. Matveev S.A., Zagidullin R.R., Smirnov A.P., et al. Parallel Numerical Algorithm for Solving Advection Equation for Coagulating Particles. Supercomputing Frontiers and Innovations. 2018. Vol. 5, no. 2. P. 43–54. DOI: 10.14529/jsfi180204.

19. McColl W.F. General Purpose Parallel Computing. Lectures on Parallel Computation, Cambridge International Series on Parallel Computation. USA, Cambridge University Press, 1993. P. 337–391.

20. Moskovsky A., Roganov V., Abramov S. Parallelism Granules Aggregation with the T-System. Parallel Computing Technologies (PaCT 2007). Lecture Notes in Computer Science. Vol. 4671. 2007. P. 293–302. DOI: 10.1007/978-3-540-73940-1_30.

21. Prifti V., Bala R., Tafa I., et al. The time profit obtained by parallelization of quicksort algorithm used for numerical sorting. Science and Information Conference (SAI). 2015. P. 897–901. DOI: 10.1109/SAI.2015.7237248.

22. Valiant L.G. A bridging model for parallel computation. Communications of the ACM. 1990. Vol. 33, no. 8. P. 103–111. DOI: 10.1145/79173.79181.

23. Voevodin V.V., Voevodin Vl.V. The V-Ray technology of optimizing programs to parallel computers. Numerical Analysis and Its Applications (WNAA 1996). Lecture Notes in Computer Science. Vol. 1196. 1997. P. 546–556. DOI: 10.1007/3-540-62598-4_136.

24. You J., Kezhang H., Liang H., et al. Research on parallel algorithms for calculating static characteristics of electromagnetic relay. IEEE 11th Conference on Industrial Electronics and Applications (ICIEA). 2016. P. 1421–1425. DOI: 10.1109/ICIEA.2016.7603808.

УДК 004.62, 004.65 DOI: 10.14529/cmse210305

ВНЕДРЕНИЕ КОНЦЕПЦИИ МАТРИЧНОГО ПРОФИЛЯ В РЕЛЯЦИОННУЮ СУБД ДЛЯ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ВРЕМЕННЫХ РЯДОВ

© 2021 Е.В. Иванова, М.Л. Цымблер

Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76)
E-mail: elena.ivanova@susu.ru, mzym@susu.ru

Поступила в редакцию: 05.07.2021

В настоящее время большие временные ряды используются в широком спектре предметных областей. Современные системы управления базами данных временных рядов (СУБД-ВР) предлагают, однако, скромный набор встроенных инструментов и средств для интеллектуального анализа данных. Использование сторонних систем интеллектуального анализа временных рядов приводит в связи с этим к нежелательным накладным расходам на экспорт данных вне СУБД-ВР, преобразование данных и импорт результатов анализа. В то же время актуальной научной задачей является внедрение методов интеллектуального анализа данных в реляционные СУБД (РСУБД), которые доминируют на рынке средств управления данными. Однако пока отсутствуют разработки по внедрению методов интеллектуального анализа временных рядов в РСУБД. В статье предлагается подход к управлению и интеллектуальному анализу временных рядов внутри РСУБД на основе концепции матричного профиля. Матричный профиль представляет собой структуру данных, которая для каждой подпоследовательности временного ряда сохраняет индекс и расстояние до ее ближайшего соседа (подпоследовательности ряда, наиболее похожей на данную). Матричный профиль служит основой для обнаружения лейтмотивов (шаблонов), аномалий и других примитивов интеллектуального анализа временных рядов. Описанный подход реализован в РСУБД PostgreSQL. Представлены результаты вычислительных экспериментов, показавшие более высокую эффективность предложенного подхода по сравнению с СУБД-ВР InfluxDB и OpenTSDB.

Ключевые слова: временные ряды, матричный профиль, PostgreSQL, InfluxDB, OpenTSDB.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Иванова Е.В., Цымблер М.Л. Внедрение концепции матричного профиля в реляционную СУБД для интеллектуального анализа временных рядов // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 3. С. 72–87. DOI: 10.14529/cmse210305.

Введение

В настоящее время, несмотря на широкое использование систем NoSQL, реляционные СУБД (РСУБД) остаются основным инструментом для хранения и обработки данных в широком спектре предметных областей, занимая до 75 % рынка СУБД [5]. РСУБД по своей природе не предоставляют встроенных функций интеллектуального анализа данных, и в связи с этим актуальной задачей является разработка методов внедрения интеллектуального анализа данных в РСУБД [15]. Действительно, если РСУБД рассматривается только как инструмент обслуживания хранилища данных, то выполнение интеллектуального анализа данных будет связано со значительными накладным расходам на экспорт больших объемов данных из РСУБД в стороннюю аналитическую систему, изменение формата данных и импорт результатов анализа обратно в РСУБД. Отметим также, что интеллектуальный анализ данных внутри РСУБД позволяет без дополнительных накладных расходов использовать системные сервисы, заложенные в архитектуре СУБД (оптимизация исполнения запросов,

целостность и безопасность данных и др.).

Методы интеллектуального анализа данных в РСУБД охватывают широкий спектр задач: поиск шаблонов [3, 21], кластеризация [17, 27], анализ графов [12, 16] и др., а также разработка соответствующих многоцелевых библиотек и фреймворков для РСУБД [6, 8, 11, 19]. Детальный обзор методов интеграции интеллектуального анализа данных в РСУБД может быть найден в работе [2].

В данной статье рассматривается проблема внедрения в РСУБД методов интеллектуального анализа временных рядов. Временной ряд представляет собой хронологическую последовательность числовых значений, отражающих течение некоторого процесса или явления. Временные ряды возникают во многих предметных областях: мониторинг показателей функциональной диагностики организма человека, моделирование климата, финансовое прогнозирование, генная инженерия и др. Однако современные системы обработки временных рядов (СУБД-ВР) предоставляют достаточно узкий спектр встроенных средств интеллектуального анализа данных (как правило, для восстановления пропусков или/и прогноза значений временного ряда) [1]. В то же время, как показал предпринятый авторами тщательный поиск научной литературы, по-видимому, пока отсутствуют разработки, обеспечивающие интеллектуальный анализ временных рядов в РСУБД.

В работе предлагается подход к управлению и интеллектуальному анализу временных рядов внутри РСУБД, который использует концепцию матричного профиля временного ряда. Матричный профиль временного ряда [23] представляет собой структуру данных, которая используется для интеллектуального анализа ряда и неформально определяется следующим образом. Матричный профиль данного временного ряда представляет собой временной ряд, i-м элементом которого является расстояние i-й подпоследовательности исходного ряда до ее ближайшего соседа (ближайшей к ней подпоследовательности исходного ряда). Матричный профиль позволяет естественным образом обнаруживать лейтмотивы (повторяющиеся шаблоны) и аномалии временного ряда как подпоследовательности ряда, которым сопоставляются минимальные и максимальные элементы профиля соответственно. Матричный профиль также служит основой для обнаружения более сложных примитивов интеллектуального анализа временных рядов: смысловые сегменты [7], эволюционирующие шаблоны [25], типичные шаблоны [9] и др.

В настоящее время концепция матричного профиля и сопутствующие алгоритмы широко используются для интеллектуального анализа временных рядов в различных приложениях цифровой индустрии. Например, в работе [24] матричные профили временных рядов энергопотребления используются для обнаружения краж электроэнергии. В работе [20] матричный профиль временного ряда синхрофазора (прибора, выполняющего синхронизированное по времени измерение параметров энергосистемы) применен для автоматического распознавания событий в энергосистеме предприятия. В [14] с помощью матричного профиля обнаруживаются периоды аномального потребления электроэнергии. В работе [10] матричный профиль временного ряда показаний датчика вибрации применяется для мониторинга рабочего состояния промышленного вентилятора. В работе [18] матричный профиль временных рядов Интернета вещей используется для реализации технического обслуживания промышленного производства.

Статья организована следующим образом. В разделе 1 приводится формальное определение матричного профиля. В разделе 2 описан подход к интеллектуальному анализу временных рядов в РСУБД на основе концепции матричного профиля. В разделе 3 пред-

ставлены результаты вычислительных экспериментов по оценке эффективности предложенного подхода. В заключении резюмированы полученные результаты и указаны направления будущих исследований.

1. Матричный профиль временного ряда

В данном разделе приводится формальное определение матричного профиля в соответствии с оригинальными работами [20, 23].

Временной ряд представляет собой хронологически упорядоченную последовательность числовых значений: $T = (t_1, \ldots, t_n), t_i \in \mathbb{R}$.

Подпоследовательность $T_{i,m}$ временного ряда T представляет собой непрерывное подмножество T, состоящее из m элементов и начинающееся с позиции i: $T_{i,m} = (t_i, \ldots, t_{i+m-1})$, где $1 \leq i \leq n - m + 1$, $1 \leq m \ll n$.

Множество всех подпоследовательностей S_T^A временного ряда T является упорядоченным набором всех возможных подпоследовательностей длины m, содержащихся в T. Подпоследовательности сортируются в порядке возрастания индекса их первого элемента: $S_T^A = \{T_{1,m}, T_{2,m}, \ldots, T_{n-m+1,m}\}$.

Профиль расстояния D_i^T представляет собой вектор расстояний между заданной подпоследовательностью $T_{i,m}$ и каждой подпоследовательностью $T_{j,m}$ из множества всех подпоследовательностей: $D_i^T = (dist(T_{i,m}, T_{1,m}), \ldots, dist(T_{i,m}, T_{n-m+1,m}))$, где $dist(\cdot, \cdot)$ означает евклидово расстояние между нормализованными подпоследовательностями.

С помощью профиля расстояния выполняется поиск ближайшего соседа для каждой подпоследовательности временного ряда, за исключением тривиальных соседей. Подпоследовательность $T_{j,m}$ называется *ближайшим соседом* подпоследовательности $T_{i,m}$, если $dist(T_{i,m}, T_{j,m}) = min(D_i^T)$. Подпоследовательность $T_{j,m}$ является *тривиальным соседом* для $T_{i,m}$, если $|i - j| \leq \frac{m}{2}$.

На основе вышеприведенных определений матричный профиль формально определяется следующим образом. *Матричный профиль* P^T временного ряда T представляет собой вектор расстояний между каждой подпоследовательностью $T_{i,m}$ и ее ближайшим нетривиальным соседом: $P^T = (nn_{nt}(D_1^T), \ldots, nn_{nt}(D_{n-m+1}^T))$, где $nn_{nt}(D_i^T)$ означает минимальное расстояние между $T_{i,m}$ и ее нетривиальными соседями.

С матричным профилем ассоциируется дополнительная структура данных для определения местоположения ближайших нетривиальных соседей. *Индекс матричного профиля* I^T временного ряда T представляет собой вектор, хранящий индекс ближайшего нетривиального соседа для каждой подпоследовательности: $I^T = (I_1^T, \ldots, I_{n-m+1}^T)$, где $I_i^T = j$, если $nn_{nt}(D_i^T) = dist(T_{i,m}, T_{j,m})$.

Матричный профиль можно рассматривать как метаданные для аннотирования соответствующего временного ряда. Например, максимальное значение профиля соответствует аномальной подпоследовательности (называемой *диссонансом* [22]), тогда как минимальные значения соответствуют наиболее похожей паре подпоследовательностей в ряде (называемой *лейтмотивом*).

Матричный профиль представляет собой агностический (не зависящий от предметной области) инструмент интеллектуального анализа временного ряда, в котором исследователь должен указать лишь один параметр — длину подпоследовательности. Среди других преимуществ матричного профиля отметим, что он может вычисляться параллельно [26] и допускает инкрементное обновление [24].

2. Внедрение концепции матричного профиля в PostgreSQL

Внедрение концепции матричного профиля в РСУБД имеет следующие основные преимущества. Выполняя интеллектуальный анализ временных рядов в РСУБД, мы избавляемся от накладных расходов на экспорт-импорт больших данных. Кроме того, будучи вычисленным и сохраненным в базе данных один раз, матричный профиль и связанные с ним примитивы интеллектуального анализа временных рядов могут затем использоваться многократно.

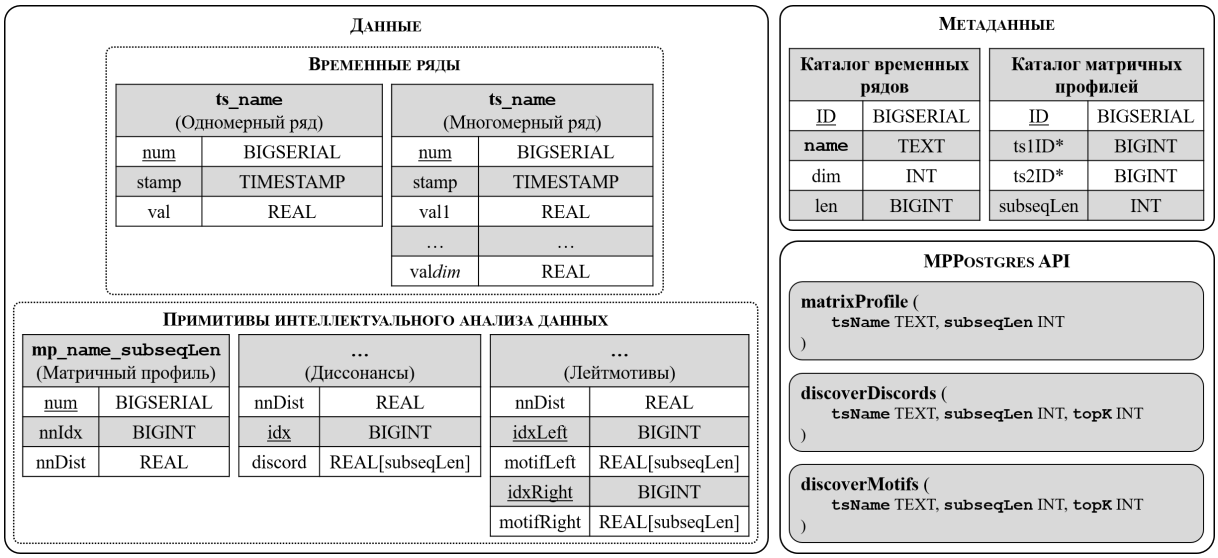


Рис. 1. Структура базы данных для управления матричным профилем в РСУБД

Внедрение матричного профиля в РСУБД предполагает создание *базы данных временных рядов*, представленной на рис. 1. Для хранения данных временных рядов предусмотрены *набор таблиц одномерных временных рядов* и *набор таблиц многомерных временных рядов*. Каждый одномерный временной ряд реализован в виде таблицы со следующими основными столбцами: индекс элемента (первичный ключ), отметка времени и собственно вещественный элемент ряда. Многомерный временной ряд реализован в виде таблицы с аналогичной структурой, где каждое измерение представлено в виде отдельного столбца с вещественными значениями.

Для хранения матричных профилей хранимых временных рядов предусмотрен *набор таблиц матричных профилей*. Каждая такая таблица хранит один матричный профиль для заданной длины подпоследовательности и содержит один вещественный столбец для хранения расстояния до ближайшего соседа и один целочисленный столбец для хранения индекса ближайшего соседа.

Помимо указанных выше таблиц, в базе данных предусмотрены следующие таблицы метаданных. В *таблице каталога временных рядов* хранится следующая основная информация о каждом временном ряде: уникальный идентификатор, имя (таблицы с данными), размерность и длина. В *таблице каталога матричных профилей* хранятся следующие основные данные о хранимых матричных профилях: уникальный идентификатор, длина подпоследовательности, внешний ключ на запись о соответствующем временном ряде в таблице каталога временных рядов.

Именование объектов базы данных временных рядов отражает связь матричного профиля с временным рядом и длиной подпоследовательности. Например, пусть данные вре-

менного ряда, получаемые с акустического датчика, требуется анализировать, используя длины подпоследовательности 64 и 128. Тогда в базе данных будут хранится таблицы `ts_acoustic`, `mp_acoustic_64` и `mp_acoustic_128` для хранения собственно временного ряда и двух его матричных профилей для соответственно.

```
1  -- Найти диссонансы временного ряда и сохранить результат в таблице
2  discoverDiscords(tsName TEXT, subseqLen INT, topK INT) RETURNS
3    TABLE (nnDist REAL, idx BIGINT, discord REAL[subseqLen])
4  -- Найти лейтмотивы временного ряда и сохранить результат в таблице
5  discoverMotifs(tsName TEXT, subseqLen INT, topK INT) RETURNS
6    TABLE (nnDist REAL, idxLeft BIGINT, motifLeft REAL[subseqLen],
7      idxRight BIGINT, motifRight REAL[subseqLen])
8  -- Вычисление матричного профиля
9  matrixProfile(tsName TEXT, subseqLen INT) RETURNS
10   TABLE (num BIGINT, nnDist REAL, nnIdx BIGINT)
```

Рис. 2. API для интеллектуального анализа данных временных рядов, MPPostgres

Помимо базы данных временных рядов, предлагаемый подход предусматривает *интерфейс прикладного программиста* (API, Application Programming Interface) для интеллектуального анализа временных рядов. API представляет собой библиотеку функций, обеспечивающих вычисление матричного профиля, обнаружение диссонансов, лейтмотивов временного ряда и др. аналитических примитивов для специфицированной длины подпоследовательности ряда и представление указанных примитивов в табличном виде (см. рис. 2). РСУБД должна поддерживать соответствующее процедурное расширение языка запросов SQL.

Диссонанс представляется в виде кортежа со следующими атрибутами: индекс диссонанса во временном ряде, расстояние до ближайшего соседа и собственно диссонанс в виде массива. Лейтмотив представляет собой кортеж со следующими атрибутами: левая и правая подпоследовательности-части лейтмотива, их индексы во временном ряде и расстояние между ними. Результат выполнения библиотечной функции может храниться в базе данных в виде таблицы или использоваться как представление (*view*, виртуальная таблица). Диссонансы и лейтмотивы вычисляются с помощью SQL-запросов, которые выбирают строки соответствующей таблицы матричного профиля с максимальными и минимальными значениями расстояния до ближайшего соседа (столбец `nnDist`) соответственно. Библиотечная функция вычисления матричного профиля реализуется как обертка над алгоритмом в оперативной памяти, предложенным в работе [26]. Указанная функция реализуется таким образом, что ее вызов осуществляется только в том случае, если соответствующая таблица матричного профиля еще не создана. API может быть дополнен функциями для поиска других примитивов интеллектуального анализа временных рядов (эволюционирующие шаблоны [25], типичные шаблоны [9] и др.).

Представленный подход был реализован нами для свободной СУБД PostgreSQL как ее расширение и получил название *MPPostgres*. Реализация выполнена на языке PL/pgSQL, который представляет собой полнофункциональный язык программирования, поддерживаемый PostgreSQL, и обеспечивает более широкий спектр процедурных возможностей, чем традиционный SQL. Примерный сценарий работы с MPPostgres показан на рис. 3: создание таблицы для хранения временного ряда показаний датчика температуры и вставка в нее данных, подключение MPPostgres к PostgreSQL, поиск диссонансов и лейтмотивов на основе матричного профиля (для длины подпоследовательности 128).

```
-- Создание таблицы временного ряда и вставка в нее данных
CREATE TABLE ts_Temperature (num BIGSERIAL, stamp TIMESTAMP, val REAL)
INSERT INTO ts_Temperature VALUES (NOW(), 21.0)
...
-- Подключение MPPostgres и вычисление матричного профиля ряда
CREATE EXTENSION MPPostgres
SELECT * FROM matrixProfile('ts_Temperature', 128)
-- Поиск диссонансов и лейтмотивов, формирование их top-100 представлений
CREATE VIEW discords_Temperature_128 AS
  SELECT * FROM discoverDiscords('ts_Temperature', 128, 100)
CREATE VIEW motifs_Temperature_128 AS
  SELECT * FROM discoverMotifs('ts_Temperature', 128, 100)
-- Вывод top-10 диссонансов и лейтмотивов
SELECT * FROM discords_Temperature_128 LIMIT 10
SELECT * FROM motifs_Temperature_128 LIMIT 10
```

Рис. 3. Пример использования MPPostgres для интеллектуального анализа временного ряда

3. Вычислительные эксперименты

Разработанный подход был реализован нами для свободной СУБД PostgreSQL 13.2 и были проведены эксперименты по исследованию его эффективности на платформе рабочей станции со следующими характеристиками: процессор Intel Xeon Gold 6254 4 ГГц, оперативная память 64 Гб, дисковая память 1 Тб. В экспериментах нами были рассмотрены два реальных приложения цифровой индустрии, связанные с анализом сенсорных данных. В указанных приложениях мы полагаем, что сенсорные данные хранятся и анализируются в PostgreSQL, и оцениваем быстродействие выполнения анализа данных. Для сравнения собственной разработки с аналогами нами также была выполнена реализация упомянутых приложений на основе СУБД-ВР InfluxDB и OpenTSDB. Конкурирующие аналоги исполняются в соответствии со следующим сценарием: сенсорные данные сперва экспортируются и преобразуются в формат, необходимый для работы стороннего аналитического приложения [4], и после выполнения им анализа данных полученные результаты импортируются в СУБД-ВР.

3.1. Выявление периодов аномального энергопотребления

В работе [14] авторы применили концепцию матричного профиля для выявления периодов аномального энергопотребления в административном, учебном и лабораторном корпусах университетского кампуса в Цюрихе (известных под псевдонимами Тревис, Трейси и Тери соответственно). Авторами использовался набор временных рядов потребления электроэнергии Building Data Genome (BDG) [13], в которых показания снимаются ежечасно. Недельным периодам аномального энергопотребления соответствуют диссонансы с длиной 168 элементов (7 дней по 24 показания).

Реализация описанного приложения в MPPostgres представлена на рис. 4. Для поиска диссонансов выполняется вызов функции разработанной библиотеки (трижды по числу корпусов кампуса, с соответствующими параметрами). Далее листинг демонстрирует способ реализации указанной функции. Строки результирующей таблицы формируются с помощью следующего цикла. Используя SQL-запрос, выполняется отбор K строк с максимальным расстоянием до подпоследовательности-ближайшего соседа из таблицы матричного профиля, после чего мы берем значения полей индекса и расстояния для строки результирующей

таблицы. Наконец, используя найденный индекс, с помощью SQL-запроса выполняется отбор элементов соответствующей подпоследовательности-диссонанса.

```
-- Найти периоды аномального энергопотребления как top-3 диссонанса
SELECT discoverDiscords('Office_Travis', 168, 3);
SELECT discoverDiscords('Classroom_Terrie', 168, 3);
SELECT discoverDiscords('Laboratory_Tracy', 168, 3);
-- Реализация функции поиска диссонансов
CREATE FUNCTION discoverDiscords(tsName TEXT, subseqLen INT, topK INT)
  RETURNS TABLE(nnDist REAL, idx BIGINT, discord REAL[subseqLen]) AS
DECLARE
  tsTabName, mpTabName TEXT;
  tsRow, mpRow RECORD;
BEGIN
  tsTabName:='ts_' || tsName; mpTabName:='mp_' || tsName || '_' || subseqLen;
  FOR mpRow IN EXEC_QUERY
    -- Получить K строк таблицы матричного профиля с max расстоянием
    SELECT * FROM mpTabName ORDER BY nnDist DESC LIMIT topK
  nnDist:=mpRow.nnDist; idx:=mpRow.num;
  FOR tsRow IN EXEC_QUERY
    -- Получить диссонанс из таблицы временного ряда
    SELECT val FROM tsTabName
    WHERE num BETWEEN idx AND idx+subseqLen-1
  discord:=array_append(discord, tsRow.val);
  RETURN NEXT ROW;
END;
```

Рис. 4. Реализация поиска периодов аномального энергопотребления в MPPostgres

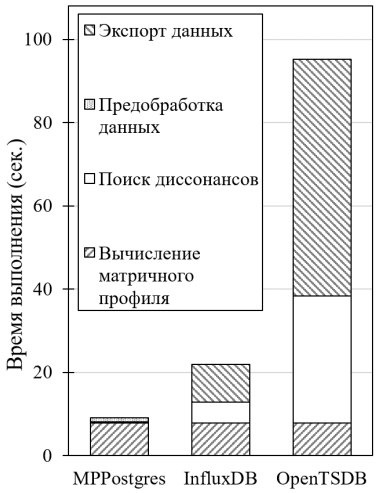


Рис. 5. Производительность различных СУБД-ВР при поиске периодов аномального энергопотребления

Результаты эксперимента представлены на рис. 5 (в иллюстративных целях набор BDG был реплицирован, чтобы соответствовать данным энергопотребления за 32 года). Можно видеть, что все конкуренты одинаково быстро вычисляют матричный профиль ряда. MPPostgres требует небольшого времени на выполнение запросов SQL, извлекающих данные для их подготовки к вычислениям. InfluxDB и OpenTSDB как решения, выполняющие

анализ данных вне СУБД-ВР, терпят значительные накладные расходы на экспорт данных, в отличие от MPPostgres. MPPostgres также превосходит конкурентов на этапе обнаружения диссонансов, поскольку использует встроенную индексацию базы данных по первичному ключу таблицы при поиске значений top-K и извлечении строк из таблиц. Следует отметить также, что при типичном сценарии, когда матричный профиль уже вычислен и сохранен в соответствующей таблице, предлагаемый подход будет еще более значительно превосходить конкурентов.

3.2. Мониторинг состояния промышленного оборудования

В работе [10] авторы применяют концепцию матричного профиля для решения задачи мониторинга промышленного вытяжного вентилятора, анализируя данные временного ряда, собранные установленным на вентиляторе датчиком вибрации. При сборе данных преднамеренно контролируется продолжительность и скорость обдува вентилятора. Переключение режимов вентилятора влияет на вибрации, в то время как лейтмотивы временного ряда указывают на моменты, когда состояние машины меняется. Поиск лейтмотивов выполняется с помощью матричного профиля ряда и вычисляется общая продолжительность временных интервалов, когда вентилятор находится в режимах «высокая скорость» и «низкая скорость», а также соотношение длительности таких интервалов. Полученные авторами результаты показывают, что соотношение, вычисленное с помощью матричного профиля, практически идентично данным натурных экспериментов.

Нами проведены эксперименты, аналогичные описанным в работе [10], поскольку ее авторы не предоставляют в отрытом доступе набор данных. Для экспериментов были собраны данные датчика вибрации, установленного на малогабаритной дробильной машине, которая находится в Южно-Уральском государственном университете и используется для подготовки инженеров. Собранный временной ряд состоит из более чем 240 тыс. элементов, соответствующих трем минутам работы машины. В ходе натурного эксперимента дробильная машина запускалась восемь раз, и статус машины менялся шестнадцать раз с «дробление остановлено» на «дробление выполняется» и обратно. Подобно работе [10], соотношение общей длительности временных интервалов, когда машина работает в разных режимах, вычисленное в наших экспериментах с помощью поиска лейтмотивов, практически совпадает с данными натурных экспериментов.

Реализация описанного случая в MPPostgres показана на рис. 6. После вычисления матричного профиля входного временного ряда используется SQL-запрос для извлечения K первых лейтмотивов в виде первых K строк таблицы матричного профиля с минимальным расстоянием до ближайшего соседа соответствующей подпоследовательности. Затем временной ряд сканируется слева направо по индексам найденных лейтмотивов, вычисляя длину интервала между предыдущим и текущим лейтмотивом, и поочередно добавляя результат к общей продолжительности, которая указывает, когда машина работает в первом или втором режиме. Наконец, после аналогичных однократных вычислений по оставшейся части входного временного ряда мы получаем результирующее соотношение.

```
1  CREATE FUNCTION trackMachineStatus(tsName TEXT, subseqLen INT, topK INT)
2    RETURNS ratio REAL[2] AS
3  DECLARE
4    status1, status2, tsLength, prevMotifIdx, i BIGINT;
5    tsTabName, mpTabName TEXT; motifRow RECORD;
6  BEGIN
7    prevMotifIdx:=0; i:=0; status1:=0; status2:=0;
8    tsTabName:='ts_' || tsName; mpTabName:='mp_' || tsName || '_' || subseqLen;
9    FOR motifRow IN EXEC_QUERY
10   -- Найти top-K лейтмотивы ряда, ...
11   SELECT * FROM (
12     SELECT num FROM _matrixProfile(tsName, subseqLen)
13     ORDER BY nnDist LIMIT topK)
14   ORDER BY num
15   -- ... сканировать их и вычислить длительность каждого статуса
16   i:=i+1;
17   IF i%2=1 THEN
18     status1:=status1+motifRow.idxLeft-prevMotifIdx;
19   ELSE
20     status2:=status2+motifRow.idxLeft-prevMotifIdx;
21   prevMotifIdx:=motifRow.idxLeft;
22   -- Вычислить длительность для остатка временного ряда, выдать результат
23   EXEC_QUERY SELECT COUNT(*) FROM tsTabName INTO tsLength;
24   IF i%2=1 THEN
25     status1:=status1+tsLength-prevMotifIdx;
26   ELSE
27     status2:=status2+tsLength-prevMotifIdx;
28   ratio[1]:=status1*100/tsLength; ratio[2]:=status2*100/tsLength;
29   RETURN ratio;
30 END;
```

Рис. 6. Реализация мониторинга состояния промышленного оборудования в MPPostgres

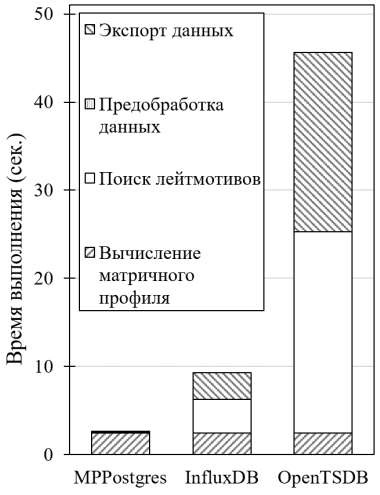


Рис. 7. Производительность различных СУБД-ВР при мониторинге состояния промышленного оборудования

На рис. 7 представлены результаты эксперимента. Как и в предыдущем случае, MPPostgres превосходит конкурентов, поскольку последние вынуждены экспортировать

данные перед их обработкой. Кроме того, в типичном сценарии, когда матричный профиль уже вычислен и сохранен в таблице матричного профиля, MPPostgres будет демонстрировать еще более высокую производительность.

Заключение

В статье представлен подход к управлению и интеллектуальному анализу временных рядов внутри реляционной СУБД (РСУБД), основанный на концепции матричного профиля временного ряда [23]. Матричный профиль представляет собой структуру данных, которая резюмирует временной ряд, сохраняя для каждой подпоследовательности временного ряда индекс и расстояние до ее ближайшего соседа (подпоследовательности ряда, наиболее похожей на данную). Матричный профиль служит основой для обнаружения лейтмотивов (шаблонов) и диссонансов (аномалий) временного ряда.

В рамках подхода данные одномерных и многомерных временных рядов хранятся в реляционных таблицах, а метаданные предоставлены таблицей каталога временных рядов. Кроме того, предусмотрены таблицы для хранения матричных профилей и каталога матричных профилей. Предложенный подход реализован в виде расширения свободной РСУБД PostgreSQL, получившего название MPPostgres. MPPostgres предоставляет прикладному программисту набор библиотечных функций для вычисления матричного профиля и следующих примитивов интеллектуального анализа временных рядов, представляющих результаты в виде реляционных таблиц: лейтмотивы и диссонансы.

Для оценки эффективности предложенного подхода нами проведены вычислительные эксперименты, использующие сценарии и данные временных рядов из двух реальных приложений цифровой индустрии: выявление периодов аномального потребления электроэнергии в зданиях и мониторинг состояния промышленного оборудования. Результаты экспериментов показали, что MPPostgres опережает по эффективности современные СУБД-ВР InfluxDB и OpenTSDB, поскольку в предложенном нами подходе отсутствуют накладные расходы на экспорт-импорт и преобразование данных.

В дальнейших исследованиях мы планируем дополнить MPPostgres более сложными примитивами интеллектуального анализа временных рядов (эволюционирующие шаблоны, типичные шаблоны и др.).

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 20-07-00140) и Министерства науки и высшего образования РФ (государственное задание FENU-2020-0022).

Литература

1. Иванова Е.В., Цымблер М.Л. Обзор современных систем обработки временных рядов // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 4. С. 79–97. DOI: 10.14529/cmse200406.

2. Цымблер М.Л. Обзор методов интеграции интеллектуального анализа данных в СУБД // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2019. Т. 8, № 2. С. 32–62. DOI: 10.14529/cmse190203.

3. Baralis E., Cerquitelli T., Chiusano S. Index Support for Frequent Itemset Mining in a Relational DBMS // Proceedings of the 21st International Conference on Data Engineering, ICDE 2005 (Tokyo, Japan, April 5–8, 2005). IEEE Computer Society, 2005. P. 754–765. DOI:

10.1109/ICDE.2005.80.

4. Benschoten A.V., Ouyang A., Bischoff F., *et al.* MPA: a novel cross-language API for time series analysis // Journal of Open Source Software. 2020. Vol. 5, no. 49. Article 2179. DOI: 10.21105/joss.02179.

5. DB-Engines Ranking of Time Series DBMS. URL: https://db-engines.com/en/ranking/time+series+dbms (дата обращения: 26.06.2021).

6. Feng X., Kumar A., Recht B., *et al.* Towards a unified architecture for in-RDBMS analytics // Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012 (Scottsdale, AZ, USA, May 20–24, 2012). P. 325–336. DOI: 10.1145/2213836.2213874.

7. Gharghabi S., Ding Y., Yeh C.M., *et al.* Matrix Profile VIII: Domain Agnostic Online Semantic Segmentation at Superhuman Performance Levels // 2017 IEEE International Conference on Data Mining, ICDM 2017 (New Orleans, LA, USA, November 18–21, 2017). P. 117–126. DOI: 10.1109/ICDM.2017.21.

8. Hellerstein J.M., Re C., Schoppmann F., *et al.* The MADlib analytics library or MAD skills, the SQL // PVLDB. 2012. Vol. 5, no. 12. P. 1700–1711. DOI: 10.14778/2367502.2367510.

9. Imani S., Madrid F., Ding W., *et al.* Matrix Profile XIII: Time Series Snippets: A New Primitive for Time Series Data Mining // 2018 IEEE International Conference on Big Knowledge, ICBK 2018 (Singapore, November 17–18, 2018). IEEE Computer Society, 2018. P. 382–389. DOI: 10.1109/ICBK.2018.00058.

10. Lee Y.Q., Beh W.L., Ooi B.Y. Tracking Operation Status of Machines through Vibration Analysis using Motif Discovery // Journal of Physics: Conference Series. 2020. Vol. 1529. Article 052005. DOI: 10.1088/1742-6596/1529/5/052005.

11. Mahajan D., Kim J.K., Sacks J., *et al.* In-RDBMS Hardware Acceleration of Advanced Analytics // Proc. VLDB Endow. 2018. Vol. 11. P. 1317–1331. DOI: 10.14778/3236187.3236188.

12. McCaffrey J.D. A Hybrid System for Analyzing Very Large Graphs // 9th International Conference on Information Technology: New Generations, ITNG 2012 (Las Vegas, Nevada, USA, 16–18 April, 2012). IEEE Computer Society, 2012. P. 253–257. DOI: 10.1109/ITNG.2012.43.

13. Miller C., Meggers F. The Building Data Genome Project: An open, public data set from non-residential building electrical meters // Energy Procedia. 2017. Vol. 122. P. 439–444. DOI: 10.1016/j.egypro.2017.07.400.

14. Nichiforov C., Stancu I., Stamatescu I., *et al.* Information Extraction Approach for Energy Time Series Modelling // 24th International Conference on System Theory, Control and Computing, ICSTCC 2020 (Sinaia, Romania, October 8–10, 2020). IEEE, 2020. P. 886–891. DOI: 10.1109/ICSTCC50638.2020.9259635.

15. Ordonez C. Can we analyze big data inside a DBMS? // Proceedings of the 16th international workshop on Data warehousing and OLAP, DOLAP 2013 (San Francisco, CA, USA, October 28, 2013). ACM, 2013. P. 85–92. DOI: 10.1145/2513190.2513198.

16. Pan C.S., Zymbler M.L. Very Large Graph Partitioning by Means of Parallel DBMS // Advances in Databases and Information Systems – 17th East European Conference, ADBIS

2013 (Genoa, Italy, September 1–4, 2013). Lecture Notes in Computer Science. Vol. 8133. Springer, 2013. P. 388–399. DOI: 10.1007/978-3-642-40683-6_29.

17. Pelekis N., Tampakis P., Vodas M., *et al.* In-DBMS Sampling-based Sub-trajectory Clustering // Proceedings of the 20th International Conference on Extending Database Technology, EDBT 2017 (Venice, Italy, March 21–24, 2017). OpenProceedings.org, 2017. P. 632–643. DOI: 10.5441/002/edbt.2017.84.

18. Pizoń J., Kulisz M., Lipski J. Matrix profile implementation perspective in Industrial Internet of Things production maintenance application // Journal of Physics: Conference Series. 2021. Vol. 1736. Article 012036. DOI: 10.1088/1742-6596/1736/1/012036.

19. Rechkalov T., Zymbler M.L. Integrating DBMS and Parallel Data Mining Algorithms for Modern Many-Core Processors // Data Analytics and Management in Data Intensive Domains – XIX International Conference, DAMDID/RCDL 2017 (Moscow, Russia, October 10–13, 2017). Communications in Computer and Information Science. Vol. 822. Springer, 2017. P. 230–245. DOI: 10.1007/978-3-319-96553-6_17.

20. Shi J., Yu N., Keogh E., *et al.* Discovering and Labeling Power System Events in Synchrophasor Data with Matrix Profile // 2019 IEEE Sustainable Power and Energy Conference (iSPEC) (Beijing, China, November 21–23, 2019). Article 19303617. DOI: 10.1109/iSPEC48194.2019.8975286.

21. Sidló C.I., Lukács A. Shaping SQL-Based Frequent Pattern Mining Algorithms // Knowledge Discovery in Inductive Databases, 4th International Workshop, KDID 2005 (Porto, Portugal, October 3, 2005), Revised Selected and Invited Papers. Lecture Notes in Computer Science. Vol. 3933. Springer, 2005. P. 188–201. DOI: 10.1007/11733492_11.

22. Yankov D., Keogh E.J., Rebbapragada U. Disk aware discord discovery: finding unusual time series in terabyte sized datasets // Knowledge and Information Systems. 2008. Vol. 17. P. 241–262. DOI: 10.1109/ICDM.2007.61.

23. Yeh C.-C.M., Zhu Y., Ulanova L., *et al.* Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile // Data Min. Knowl. Discov. 2018. Vol. 32, no. 1. P. 83–123. DOI: 10.1007/s10618-017-0519-9.

24. Zhu Y., Gharghabi S., Silva D.F., *et al.* The Swiss army knife of time series data mining: ten useful things you can do with the matrix profile and ten lines of code // Data Mining and Knowledge Discovery. 2020. Vol. 34. P. 949–979. DOI: 10.1007/s10618-019-00668-6.

25. Zhu Y., Imamura M., Nikovski D., *et al.* Matrix Profile VII: Time Series Chains: A New Primitive for Time Series Data Mining // 2017 IEEE International Conference on Data Mining, ICDM 2017 (New Orleans, LA, USA, November 18–21, 2017). P. 695–704. DOI: 10.1109/ICDM.2017.79.

26. Zhu Y., Zimmerman Z., Senobari N.S., *et al.* Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins // IEEE 16th International Conference on Data Mining, ICDM 2016 (Barcelona, Spain, December 12–15, 2016). IEEE Computer Society, 2016. P. 739–748. DOI: 10.1109/ICDM.2016.0085.

27. Zymbler M.L., Kraeva Y., Grents A., *et al.* An Approach to Fuzzy Clustering of Big Data Inside a Parallel Relational DBMS // Data Analytics and Management in Data Intensive Domains – 21st International Conference, DAMDID/RCDL 2019 (Kazan, Russia, October 15–

18, 2019). Communications in Computer and Information Science. Vol. 1223. Springer, 2019. P. 211–223. DOI: 10.1007/978-3-030-51913-1_14.

Иванова Елена Владимировна, к.ф.-м.н., кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Цымблер Михаил Леонидович, д.ф.-м.н., доцент, кафедра системного программирования, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

DOI: 10.14529/cmse210305

EMBEDDING OF THE MATRIX PROFILE CONCEPT INTO A RELATIONAL DBMS FOR TIME SERIES MINING

© 2021 E.V. Ivanova, M.L. Zymbler

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia)

E-mail: elena.ivanova@susu.ru, mzym@susu.ru

Received: 05.07.2021

Currently, large time series are used in a wide range of subject areas. Modern time series DBMSs (TSDBMS) offer, however, a modest set of built-in tools for data mining. The use of third-party time series mining systems to undesirable overhead costs for exporting data outside the TSDBMS, converting data and importing analysis results. At the same time, there is a topical issue of the embedding of data mining methods into relational DBMSs (RDBMS), which dominate the market of data management tools. However, there are still no developments of time series mining methods in RDBMS. The article proposes an approach to the management and mining of time series data within the RDBMS based on the matrix profile concept. A matrix profile is a data structure that, for each subsequence of a time series, stores the index of and the distance to its nearest neighbor. The matrix profile serves as the basis for detecting motifs, anomalies and other primitives of time series mining. The proposed approach is implemented in the PostgreSQL RDBMS. The experimental results showed a higher efficiency of the proposed approach compared to the TSDBMS InfluxDB and OpenTSDB.

Keywords: time series, matrix profile, PostgreSQL, InfluxDB, OpenTSDB.

FOR CITATION

Ivanova E.V., Zymbler M.L. Embedding of the Matrix Profile Concept Into a Relational DBMS for Time Series Mining. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2021. Vol. 10, no. 3. P. 72–87. (in Russian) DOI: 10.14529/cmse210305.

References

1. Ivanova E.V., Zymbler M.L. Overview of Modern Time Series Management Systems. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2020. Vol. 9, no. 4. P. 79–97. DOI: 10.14529/cmse200406. (in Russian)

2. Zymbler M.L. Overview of Methods for Integrating Data Mining into DBMS. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.

2019. Vol. 8, no. 2. P. 32–62. DOI: 10.14529/cmse190203. (in Russian)

3. Baralis E., Cerquitelli T., Chiusano S. Index Support for Frequent Itemset Mining in a Relational DBMS. Proceedings of the 21st International Conference on Data Engineering, ICDE 2005 (Tokyo, Japan, April 5–8, 2005). IEEE Computer Society, 2005. P. 754–765. DOI: 10.1109/ICDE.2005.80.

4. Benschoten A.V., Ouyang A., Bischoff F., *et al.* MPA: a novel cross-language API for time series analysis. Journal of Open Source Software. 2020. Vol. 5, no. 49. Article 2179. DOI: 10.21105/joss.02179.

5. DB-Engines Ranking of Time Series DBMS. URL: https://db-engines.com/en/ranking/time+series+dbms Available at: `https://docs.timescale.com/` (accessed: 26.06.2021).

6. Feng X., Kumar A., Recht B., *et al.* Towards a unified architecture for in-RDBMS analytics. Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012 (Scottsdale, AZ, USA, May 20–24, 2012). P. 325–336. DOI: 10.1145/2213836.2213874.

7. Gharghabi S., Ding Y., Yeh C.M., *et al.* Matrix Profile VIII: Domain Agnostic Online Semantic Segmentation at Superhuman Performance Levels. 2017 IEEE International Conference on Data Mining, ICDM 2017 (New Orleans, LA, USA, November 18–21, 2017). P. 117–126. DOI: 10.1109/ICDM.2017.21.

8. Hellerstein J.M., Re C., Schoppmann F., *et al.* The MADlib analytics library or MAD skills, the SQL. PVLDB. 2012. Vol. 5, no. 12. P. 1700–1711. DOI: 10.14778/2367502.2367510.

9. Imani S., Madrid F., Ding W., *et al.* Matrix Profile XIII: Time Series Snippets: A New Primitive for Time Series Data Mining. 2018 IEEE International Conference on Big Knowledge, ICBK 2018 (Singapore, November 17–18, 2018). IEEE Computer Society, 2018. P. 382–389. DOI: 10.1109/ICBK.2018.00058.

10. Lee Y.Q., Beh W.L., Ooi B.Y. Tracking Operation Status of Machines through Vibration Analysis using Motif Discovery. Journal of Physics: Conference Series. 2020. Vol. 1529. Article 052005. DOI: 10.1088/1742-6596/1529/5/052005.

11. Mahajan D., Kim J.K., Sacks J., *et al.* In-RDBMS Hardware Acceleration of Advanced Analytics. Proc. VLDB Endow. 2018. Vol. 11. P. 1317–1331. DOI: 10.14778/3236187.3236188.

12. McCaffrey J.D. A Hybrid System for Analyzing Very Large Graphs. 9th International Conference on Information Technology: New Generations, ITNG 2012 (Las Vegas, Nevada, USA, April 16–18, 2012). IEEE Computer Society, 2012. P. 253–257. DOI: 10.1109/ITNG.2012.43.

13. Miller C., Meggers F. The Building Data Genome Project: An open, public data set from non-residential building electrical meters. Energy Procedia. 2017. Vol. 122. P. 439–444. DOI: 10.1016/j.egypro.2017.07.400.

14. Nichiforov C., Stancu I., Stamatescu I., *et al.* Information Extraction Approach for Energy Time Series Modelling. 24th International Conference on System Theory, Control and Computing, ICSTCC 2020 (Sinaia, Romania, October 8–10, 2020). IEEE, 2020. P. 886–891. DOI: 10.1109/ICSTCC50638.2020.9259635.

15. Ordonez C. Can we analyze big data inside a DBMS? Proceedings of the 16th international workshop on Data warehousing and OLAP, DOLAP 2013 (San Francisco, CA, USA, October 28, 2013). ACM, 2013. P. 85–92. DOI: 10.1145/2513190.2513198.

16. Pan C.S., Zymbler M.L. Very Large Graph Partitioning by Means of Parallel DBMS. Advances in Databases and Information Systems – 17th East European Conference, ADBIS 2013 (Genoa, Italy, September 1–4, 2013). Lecture Notes in Computer Science. Vol. 8133. Springer, 2013. P. 388–399. DOI: 10.1007/978-3-642-40683-6_29.

17. Pelekis N., Tampakis P., Vodas M., *et al.* In-DBMS Sampling-based Sub-trajectory Clustering. Proceedings of the 20th International Conference on Extending Database Technology, EDBT 2017 (Venice, Italy, March 21–24, 2017). OpenProceedings.org, 2017. P. 632–643. DOI: 10.5441/002/edbt.2017.84.

18. Pizoń J., Kulisz M., Lipski J. Matrix profile implementation perspective in Industrial Internet of Things production maintenance application. Journal of Physics: Conference Series. 2021. Vol. 1736. Article 012036. DOI: 10.1088/1742-6596/1736/1/012036.

19. Rechkalov T., Zymbler M.L. Integrating DBMS and Parallel Data Mining Algorithms for Modern Many-Core Processors. Data Analytics and Management in Data Intensive Domains – XIX International Conference, DAMDID/RCDL 2017 (Moscow, Russia, October 10–13, 2017). Communications in Computer and Information Science. Vol. 822. Springer, 2017. P. 230–245. DOI: 10.1007/978-3-319-96553-6_17.

20. Shi J., Yu N., Keogh E., *et al.* Discovering and Labeling Power System Events in Synchrophasor Data with Matrix Profile. 2019 IEEE Sustainable Power and Energy Conference (iSPEC) (Beijing, China, November 21–23, 2019). Article 19303617. DOI: 10.1109/iSPEC48194.2019.8975286.

21. Sidló C.I., Lukács A. Shaping SQL-Based Frequent Pattern Mining Algorithms. Knowledge Discovery in Inductive Databases, 4th International Workshop, KDID 2005 (Porto, Portugal, October 3, 2005), Revised Selected and Invited Papers. Lecture Notes in Computer Science. Vol. 3933. Springer, 2005. P. 188–201. DOI: 10.1007/11733492_11.

22. Yankov D., Keogh E.J., Rebbapragada U. Disk aware discord discovery: finding unusual time series in terabyte sized datasets. Knowledge and Information Systems. 2008. Vol. 17. P. 241–262. DOI: 10.1109/ICDM.2007.61.

23. Yeh C.-C.M., Zhu Y., Ulanova L., *et al.* Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile. Data Min. Knowl. Discov. 2018. Vol. 32, no. 1. P. 83–123. DOI: 10.1007/s10618-017-0519-9.

24. Zhu Y., Gharghabi S., Silva D.F., *et al.* The Swiss army knife of time series data mining: ten useful things you can do with the matrix profile and ten lines of code. Data Mining and Knowledge Discovery. 2020. Vol. 34. P. 949–979. DOI: 10.1007/s10618-019-00668-6.

25. Zhu Y., Imamura M., Nikovski D., *et al.* Matrix Profile VII: Time Series Chains: A New Primitive for Time Series Data Mining. 2017 IEEE International Conference on Data Mining, ICDM 2017 (New Orleans, LA, USA, November 18–21, 2017). P. 695–704. DOI: 10.1109/ICDM.2017.79.

26. Zhu Y., Zimmerman Z., Senobari N.S., *et al.* Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins. IEEE

16th International Conference on Data Mining, ICDM 2016 (Barcelona, Spain, December 12–15, 2016). IEEE Computer Society, 2016. P. 739–748. DOI: 10.1109/ICDM.2016.0085.

27. Zymbler M.L., Kraeva Y., Grents A., *et al.* An Approach to Fuzzy Clustering of Big Data Inside a Parallel Relational DBMS. Data Analytics and Management in Data Intensive Domains – 21st International Conference, DAMDID/RCDL 2019 (Kazan, Russia, October 15–18, 2019). Communications in Computer and Information Science. Vol. 1223. Springer, 2019. P. 211–223. DOI: 10.1007/978-3-030-51913-1_14.

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 10, № 3
2021

16+

Техн. редактор *А.В. Миних*

Издательский центр Южно-Уральского государственного университета

Подписано в печать 31.08.2021. Дата выхода в свет 15.09.2021. Формат 60×84 1/8. Печать цифровая.
Усл. печ. л. 10,69. Тираж 500 экз. Заказ 285/325. Цена свободная.

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