ISSN 1561-2430 (Print)
ISSN 2524-2415 (Online)
УДК 519.671

Поступила в редакцию 21.12.2017
Received 21.12.2017

Н. А. Лиходед, А. А. Толстиков

Белорусский государственный университет, Минск, Беларусь

ФОРМАЛИЗОВАННОЕ ПОЛУЧЕНИЕ КОММУНИКАЦИОННЫХ ОПЕРАЦИЙ ПАРАЛЛЕЛЬНЫХ ЗЕРНИСТЫХ АЛГОРИТМОВ

Аннотация. Алгоритмы, предназначенные для реализации на параллельных компьютерах с распределенной памятью, включают в себя вычислительные макрооперации (зерна вычислений), а также коммуникационные операции, которые в явном виде задают обмен массивами данных между вычислительными узлами. Наибольшие затруднения вызывает, как правило, задача организации обмена данными. Для ее решения надо сначала выявить информационные зависимости между макрооперациями, а затем сгенерировать порождаемые этими зависимостями коммуникационные операции. Для автоматизации и упрощения процесса генерации кода необходима формализация получения коммуникационных операций. Такая формализация известна для случая однородных информационных зависимостей. Она использует представление зависимостей между макрооперациями векторами глобальных зависимостей. Известны также результаты, которые позволяют получить пересылаемые массивы данных, но при этом требуют применения инструментальных средств для работы с многогранниками и не формализуют коммуникационные операции. В настоящем исследовании представлен способ формализации и включения в структуру алгоритма коммуникационных операций получения и отправки массивов данных в параллельном алгоритме с аффинными зависимостями. Применение функций, определяющих зависимости между макрооперациями, позволило в алгоритме, задающем параллельные вычислительные процессы, получить явные представления коммуникационных операций. Исследования являются обобщением формализации операций пересылки элементов массивов в параллельном алгоритме, операции которого не разбиты на макрооперации, а также некоторых аспектов метода получения коммуникационных операций, определяемых однородными информационными зависимостями.

Ключевые слова: параллельные вычисления, распараллеливание алгоритмов, параллельный компьютер с распределенной памятью, формализация коммуникационных операций

Для цитирования. Лиходед, Н. А. Формализованное получение коммуникационных операций параллельных зернистых алгоритмов / Н. А. Лиходед, А. А. Толстиков // Вес. Нац. акад. навук Беларусі. Сер. фіз.-мат. навук. – 2018. – Т. 54, № 1. – С. 50–61.

N. A. Likhoded, A. A. Tolstsikau

Belarusian State University, Minsk, Belarus

FORMALIZED OBTAINING OF THE COMMUNICATION OPERATIONS OF GRAINED PARALLEL ALGORITHMS

Abstract. Algorithms designed for implementation on parallel computers with distributed memory consist of computational macro operations (calculation grains) and communication operations specifying the data arrays exchange between computing nodes. The major difficulty is how to find an efficient way to organize the data exchange. To solve this problem, it is first necessary to identify information dependences between macro operations and then to generate the communication operations caused by these dependences. To automate and simplify the process of code generation, it is necessary to formalize communication operations. The formalization is known for the case of homogeneous information dependences. Such formalization uses the vectors of global dependences as a representation of dependences between the calculation grains. Also, there is a way that makes it possible to obtain the data arrays exchange, but it requires the usage of tools to work with polyhedra and does not formalize communication operations. This article presents a formalization method and a method of inclusion of communication operations into the algorithm structure (receiving and sending data arrays) in case of a parallel algorithm with affine dependences. The usage of functions determining the relationship between macro operations allowed obtaining explicit representations of communication operations. This work is a generalization of the formalization of the operations of sending data in a parallel algorithm, where operations are not divided into macro operations, as well as a generalization of some aspects of obtaining the communication operation method.

Keywords: parallel computations, parallelization of algorithms, distributed memory parallel computer, formalization of communication operations

For citation. Likhoded N. A., Tolstsikau A. A. Formalized obtaining of the communication operations of grained parallel algorithms. *Vestsi Natsyianal'nai akademii navuk Belarusi. Seryia fizika-matematychnykh navuk = Proceedings of the National Academy of Sciences of Belarus. Physics and Mathematics series*, 2018, vol. 54, no. 1, pp. 50–61 (in Russian).

Введение. При получении параллельных алгоритмов для компьютеров с распределенной памятью требуется в явном виде указать операции обмена данными. Это новые операции, в исходной записи указаны только вычислительные операции. Для алгоритмов, реализуемых на параллельных компьютерах с распределенной памятью, характерна зернистая структура: множество операций алгоритма разбито на макрооперации, называемые зернами вычислений или тайлами [1–3]; операции одного тайла выполняются атомарно, как одна единица вычислений. Результаты вычислений тайла (как правило, массив) передаются, если требуется обмен, одной коммуникационной операцией, что позволяет значительно уменьшить общее время пересылки данных. Коммуникационные операции зернистого алгоритма порождаются информационными зависимостями между тайлами, а также входными и выходными данными. Для получения зерен (блоков) вычислений можно использовать не только тайлинг, но и другие преобразования. В частности, на практике применяются разбиения матриц по строкам и/или по столбцам.

Генерируют коммуникационные операции обычно «вручную», для чего требуется детальное понимание алгоритма. Для автоматизации и упрощения процесса генерации кода необходима теория получения коммуникационных операций. Отметим результаты, связанные с задачей включения коммуникационных операций в параллельный зернистый алгоритм. В работе [4] формализованы операции пересылки элементов массивов в параллельном алгоритме, операции которого не разбиты на тайлы. В [5] задача решена для коммуникационных операций, определяемых однородными информационными зависимостями; использовано представление зависимостей между тайлами векторами глобальных (уровня тайлов) зависимостей [6]. В [7, 8] выделяются данные, которые необходимо пересылать между процессорами, но коммуникационные операции не формализованы; для получения пересылаемых массивов данных следует применять инструментальные средства для работы с многогранниками; приведенные в работах алгоритмы реализованы в открытой системе PLUTO [9].

Отметим важность коммуникационных операций и в том случае, если их не надо записывать в явном виде. Имеются в виду многоядерные CPU (как при последовательной, так и параллельной реализации) и GPU. Время решения задачи на современном компьютере определяется не столько скоростью выполнения вычислительных операций, сколько скоростью доступа к памяти («стена памяти» – примерно 100 тактов при обращении процессора к оперативной памяти). Скорость доступа существенно зависит от места в иерархической памяти. Время считывания данных из оперативной памяти многократно (например, в 15 раз) превосходит время считывания из кэш-памяти.

Таким образом, объем и сложность коммуникационных операций является одним из важнейших параллельных свойств алгоритма. В параллельных вычислениях наряду с традиционными вычислительными свойствами алгоритма – общее число выполняемых операций, влияние ошибок округления, размер необходимой памяти, сходимость – требуется исследовать новые, параллельные свойства [10]: число параллельных множеств, высота и ширина параллельной формы алгоритма, число параллельных вычислительных процессов, объем коммуникационных операций, сложность коммуникационных связей.

В этой работе предложен метод формализации коммуникационных операций получения и отправки массивов данных в параллельном зернистом алгоритме с аффинными зависимостями. Использование функций, определяющих зависимости между тайлами [11], позволило получить явные представления коммуникационных операций зернистых вычислений.

1. Предварительные сведения. Приведем некоторые сведения о рассматриваемом классе алгоритмов, информационных зависимостях между операциями алгоритма, получении макроопераций (тайлинг).

Будем считать, что алгоритм задан последовательной программой, основную вычислительную часть которой составляет гнездо циклов произвольной структуры вложенности, а границы изменения параметров циклов задаются аффинными функциями (неоднородными формами, линейными по совокупности параметров циклов и внешних переменных). Такие алгоритмы называют алгоритмами с аффинными зависимостями, или последовательными программами линейного класса [12]. Пусть в гнезде циклов имеется K выполняемых операторов S_β. Область изменения параметров циклов для оператора S_β и размерность этой области обозначим соответственно V_β и n_β, $1 \le \beta \le K$.

Вхождением (a,S_β,q) будем называть q-е вхождение массива a в оператор S_β. Нахождение значения оператора S_β при конкретных значениях β и вектора параметров цикла J будем называть операцией и обозначать $S_\beta(J)$. Индексы элементов массива, связанных с вхождением (a,S_β,q), выражаются функцией доступа $\overline{F}_{a,S_\beta,q}(J)$.

Пусть пара вхождений $(a,S_\alpha,1)$ и (a,S_β,q) порождает истинную зависимость $S_\alpha(I)\rightarrow S_\beta(J)$. Это означает: $S_\alpha(I)$ выполняется раньше $S_\beta(J)$; $S_\alpha(I)$ переопределяет (изменяет) элемент массива a, а $S_\beta(J)$ использует в качестве аргумента тот же самый элемент массива; между операциями $S_\alpha(I)$ и $S_\beta(J)$ этот элемент не переопределяется. Наглядным представлением зависимостей (информационных связей) является граф алгоритма [12] – ориентированный граф, в котором вершины порождаются операциями алгоритма, а дуги – зависимостями между операциями.

Зависимости между операциями можно задать функциями зависимостей $\overline{\Phi}_{\alpha,\beta}(J)$. Функция $\overline{\Phi}_{\alpha,\beta}(J)$ позволяет для операции $S_\beta(J)$ найти операцию $S_\alpha(I)$, от которой зависит $S_\beta(J)$. Функции зависимостей, называемые также покрывающими функциями графа алгоритма, являются удобным математическим аппаратом для описания информационных связей между операциями алгоритма [12, 13]. Обычно рассматривают аффинные функции зависимостей:

$$\overline{\Phi}_{\alpha,\beta}(J) = \Phi_{\alpha,\beta}J + \Psi_{\alpha,\beta}N - \varphi^{\alpha,\beta}, \quad J \in V_{\alpha,\beta}, \tag{1}$$

где $\Phi_{\alpha,\beta}$ и $\Psi_{\alpha,\beta}$ – матрицы, $\varphi^{\alpha,\beta}$ – вектор, N – вектор внешних переменных алгоритма, $V_{\alpha,\beta}$ – область определения функции.

Пусть в гнезде циклов имеется Θ наборов выполняемых операторов. Под набором операторов будем понимать один или несколько операторов, окруженных одним и тем же множеством циклов. Операторы и наборы операторов линейно упорядочены их расположением в записи алгоритма. Обозначим V^θ, $1 \le \theta \le \Theta$, – области изменения параметров циклов, окружающих наборы операторов; n^θ – размерность области V^θ, число циклов, окружающих θ-й набор операторов. Будем считать, что если оператор S_β принадлежит набору операторов с номером θ^β, то $n^{\theta^\beta} = n_\beta$.

Тайлинг применяется для получения макроопераций, называемых зернами вычислений, или тайлами. Тайлинг представляет собой преобразование алгоритма, при котором 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. Согласно тайлингу для каждого набора операторов локальные циклы должны быть самыми внутренними. Поэтому общие локальные циклы распределяются между циклами каждого набора операторов; локальные циклы переставляются с глобальными и становятся самыми внутренними.

Отметим, что выделение макроопераций-тайлов (блоков вычислений) применяется не только для увеличения зернистости при реализации алгоритмов на параллельных компьютерах с распределенной памятью, но и для эффективного использования иерархической памяти при реализации (как последовательной, так и параллельной) на CPU, а также для организации вычислений на GPU.

Для формализации тайлинга используются следующие величины и множества:

$m_\zeta^\theta = \min\limits_{J\left(j_1,j_2,\ldots,j_{n^\theta}\right)\in V^\theta} j_\zeta$, $M_\zeta^\theta = \max\limits_{J\left(j_1,j_2,\ldots,j_{n^\theta}\right)\in V^\theta} j_\zeta$, $1 \le \zeta \le n^\theta$, – предельные значения изменения параметров циклов;

$r_1^\theta,...,r_{n^\theta}^\theta$ – заданные натуральные числа, определяющие размеры тайла; r_ζ^θ обозначает число значений параметра j_ζ, приходящихся на один тайл θ-го набора операторов; если два набора операторов имеют общий цикл с параметром j_ζ, то $r_\zeta^{\theta_1} = r_\zeta^{\theta_2}$;

$Q_\zeta^\theta = \left\lceil (M_\zeta^\theta - m_\zeta^\theta + 1) / r_\zeta^\theta \right\rceil, 1 \le \zeta \le n^\theta,$ – число частей, на которые при формировании тайлов разбивается область значений параметра j_ζ цикла, окружающего θ-й набор операторов;

$V^{\theta,gl} = \left\{ J^{gl}(j_1^{gl},...,j_{n^\theta}^{gl}) \middle| \, 0 \le j_\zeta^{gl} \le Q_\zeta^\theta - 1, \ 1 \le \zeta \le n^\theta \right\}$ – области изменения глобальных параметров, т. е. уровня тайлов, циклов;

$V_{J^{gl}}^\theta = \left\{ J(j_1,...,j_{n^\theta}) \in V^\theta \middle| \, m_\zeta^\theta + j_\zeta^{gl} r_\zeta^\theta \le j_\zeta \le m_\zeta^\theta - 1 + (j_\zeta^{gl}+1) r_\zeta^\theta, \ 1 \le \zeta \le n^\theta \right\}, \ J^{gl} \in V^{\theta,gl},$ – области изменения параметров локальных (уровня операций тайлов) циклов при фиксированных значениях параметров глобальных циклов. Множество операций, выполняемых на итерациях множества $V_{J^{gl}}^\theta$, также обозначают $V_{J^{gl}}^\theta$. Множества $V_{J^{gl}}^\theta$ называются тайлами $V_{J^{gl}}^\theta$, или тайлами J^{gl}. Тайлы $V_{J^{gl}}^\theta$ будем называть тайлами θ-го типа.

Опишем структуру многомерного цикла после применения к нему преобразования тайлинга. Параметры циклов j_ζ^{gl} изменяются в соответствии с неравенствами $0 \le j_\zeta^{gl} \le Q_\zeta^\theta$, для каждого набора операторов имеется столько локальных циклов, сколько $r_1^\theta,...,r_{n^\theta}^\theta$ превосходят единицу.

Пример. Пусть A – левая треугольная матрица порядка n с диагональными элементами, равными единице, b – n-мерный вектор. Рассмотрим алгоритм решения системы $Ax = b$ методом обратной подстановки:

```
S1: x(1) = b(1)
do i = 2, n
    S2: x(i) = b(i,)
    do j = 1, i – 1                                    (2)
        S3: x(i) = x(i) – a(i,j)x(j)
    enddo
enddo
```

После выделения макроопераций-тайлов (блочный относительно $a(i,j)$ алгоритм с числом блоков Q_1 по i и Q_2 по j, блоки размера $r_1 \times r_2$) получим

```
S1: x(1) = b(1)
do i^gl = 0, Q1 – 1 // Одна итерация цикла – «обработка» r1 строк матрицы A
    // Начало тайла Tile1(i^gl)
    do i = 2 + i^gl r1, min(1 + (i^gl + 1)r1, n)
        S2: x(i) = b(i)
    enddo
    // Конец тайла Tile1(i^gl)
    do j^gl = 0, Q2 – 1 // Одна итерация цикла – «обработка» r2 столбцов матрицы A
        // Начало тайла Tile2(i^gl,j^gl)
        do i = 2 + i^gl r1, min(1 + (i^gl + 1)r1, n) //
            do j = 1 + j^gl r2, min((j^gl + 1)r2, i – 1)
                S3: x(i) = x(i) – a(i,j)x(j)
            enddo
        enddo
        // Конец тайла Tile2(i^gl,j^gl)
    enddo
enddo
```

Граф рассматриваемого алгоритма с очертаниями тайлов изображен на рис. 1.

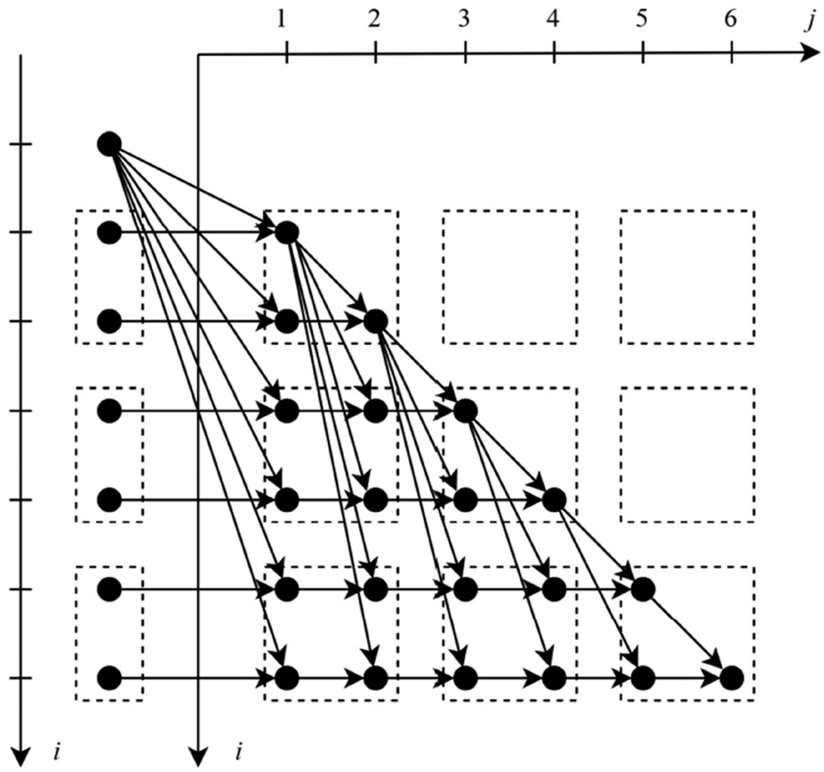

Рис. 1. Граф алгоритма (2) с очертаниями тайлов (n = 7, $r_1 = r_2$ = 2)
Fig. 1. Graph of algorithm (2) with shaped tiles (n = 7, $r_1 = r_2$ = 2)

Приведем функцию $\overline{\Phi}_{3,3}$ ([12], с. 381) для аффинной зависимости, порождаемой вхождением $(x,S_3,1)$ (т. е. вхождением $x\left(\overline{F}_{x,S3,1}(i,j)\right)=x(i)$) и вхождением $(x,S_3,3)$ (вхождением $x\left(\overline{F}_{x,S3,3}(i,j)\right)=x(j)$):

$$\overline{\Phi}_{3,3}\left(i,j\right)=\begin{pmatrix}0&1\\0&1\end{pmatrix}\begin{pmatrix}i\\j\end{pmatrix}-\begin{pmatrix}0\\1\end{pmatrix}=\left(j,j-1\right),\ \left(i,j\right)=\left\{(i,j)\in Z^2\,|3\le i\le n,2\le j\le i-1\right\}. \qquad (3)$$

2. Функции глобальных зависимостей. Рассмотрим некоторую истинную зависимость $S_\alpha(I)\to S_\beta(J)$. Итерации, порождающие зависимость, принадлежат некоторым тайлам: $I\in V_{I^{gl}}^{\theta^\alpha}$ и $J\in V_{J^{gl}}^{\theta^\beta}$. Таким образом, порождается зависимость уровня тайлов: данное, вычисленное при выполнении операций тайла $V_{I^{gl}}^{\theta^\alpha}$, используется при выполнении операции (операций) тайла $V_{J^{gl}}^{\theta^\beta}$. По аналогии с функциями зависимостей $\overline{\Phi}_{\alpha,\beta}(J)$ уровня операций введем в рассмотрение функции $\overline{\Phi}_{\alpha,\beta}^{gl}\left(J^{gl}\right)$, которые для любого тайла $V_{J^{gl}}^{\theta^\beta}$ определяют тайлы $V_{I^{gl}}^{\theta^\alpha}$, $I^{gl}=\overline{\Phi}_{\alpha,\beta}^{gl}\left(J^{gl}\right)$, результаты операций которых являются аргументами операций тайла $V_{J^{gl}}^{\theta^\beta}$. Область определения функции $\overline{\Phi}_{\alpha,\beta}^{gl}\left(J^{gl}\right)$ обозначим $V_{\alpha,\beta}^{gl}$.

Доказано [11], что функция глобальных зависимостей имеет вид

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

где матрицы $\Phi_{\alpha,\beta}^{gl}$, $\Psi_{\alpha,\beta}^{gl}$ и векторы $\varphi^{\alpha,\beta,gl_m}$ выражаются через параметры тайлинга и функции зависимостей (1). Одна функция зависимостей $\overline{\Phi}_{\alpha,\beta}(J)$ может порождать только одну матрицу

$\Phi^{gl}_{\alpha,\beta}$ и одну матрицу $\Psi^{gl}_{\alpha,\beta}$, но, вообще говоря, несколько векторов $\varphi^{\alpha,\beta,gl_m}$ и, соответственно, несколько функций глобальных зависимостей $\overline{\Phi}^{gl_m}_{\alpha,\beta}$.

Формализуем множества информационно зависимых операций тайлов, порождающие глобальные зависимости:

$V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}$ – подмножество итераций тайла $V^{\theta^\beta}_{J^{gl}}$, при выполнении которых на вхождении (a,S_β,q) используются результаты операций тайла $V^{\theta^\alpha}_{I^{gl}}$, $I^{gl}=\overline{\Phi}^{gl_m}_{\alpha,\beta}\left(J^{gl}\right)$, полученные на вхождении $(a,S_\alpha,1)$;

$V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}$ – подмножество итераций тайла $V^{\theta^\alpha}_{I^{gl}}$, при выполнении которых результаты операций, полученные на вхождении $(a,S_\alpha,1)$, используются в качестве входных данных на вхождении (a,S_β,q) для операций тайла (тайлов) $V^{\theta^\beta}_{J^{gl}}$.

Для получения $V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}$ и $V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}$ требуется найти в параметризованном виде решения систем уравнений и неравенств:

$$V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}=\left\{J\in V^{\theta^\beta}_{J^{gl}}\;\middle|\;\overline{\Phi}_{\alpha,\beta}(J)\in V^{\theta^\alpha}_{I^{gl}},\; I^{gl}=\overline{\Phi}^{gl_m}_{\alpha,\beta}\left(J^{gl}\right)\right\}, \tag{4}$$

$$V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}=\left\{I\in V^{\theta^\alpha}_{I^{gl}}\;\middle|\;I=\overline{\Phi}_{\alpha,\beta}(J)\in V^{\theta^\alpha}_{I^{gl}},\; J\in V^{\theta^\beta}_{J^{gl}},\; I^{gl}=\overline{\Phi}^{gl_m}_{\alpha,\beta}\left(J^{gl}\right),\; J^{gl}\in V^{gl_m}_{\alpha,\beta}\right\}. \tag{5}$$

Для получения $V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}$ требуется, как правило, применить еще процедуру исключения Фурье – Моцкина (Fourier – Motzkin elimination – процедура, предназначенная для выделения из системы неравенств ограничений, связанных только с интересующими нас переменными).

П р и м е р (продолжение). Функция аффинных зависимостей (3) порождает в случае $r_1=r_2=r$ две функции глобальных зависимостей [11]:

$$\overline{\Phi}^{gl_m}_{3,3}(i^{gl},j^{gl})=\begin{pmatrix}0&1\\0&1\end{pmatrix}\begin{pmatrix}i^{gl}\\j^{gl}\end{pmatrix}-\varphi^{3,3,gl_m},\quad \varphi^{3,3,gl_1}=\begin{pmatrix}0\\0\end{pmatrix},\quad \varphi^{3,3,gl_2}=\begin{pmatrix}1\\1\end{pmatrix}, \tag{6}$$

$$V^{gl_1}_{\alpha,\beta}=\left\{(i^{gl},j^{gl})\in Z^2\;|\;1\le j^{gl}<i^{gl}\le Q_1-1\right\},\; V^{gl_2}_{\alpha,\beta}=\left\{(i^{gl},j^{gl})\in Z^2\;|\;1<j^{gl}<i^{gl}\le Q_1-1\right\}.$$

На рис. 2 изображен граф зависимостей уровня тайлов, порождаемых функциями глобальных зависимостей $\overline{\Phi}^{gl_1}_{3,3}$, $\overline{\Phi}^{gl_2}_{3,3}$, а также вектором глобальных зависимостей $\varphi^{gl}=(0,1)$, порождаемых однородными зависимостями между операциями $S_3(i,j)$.

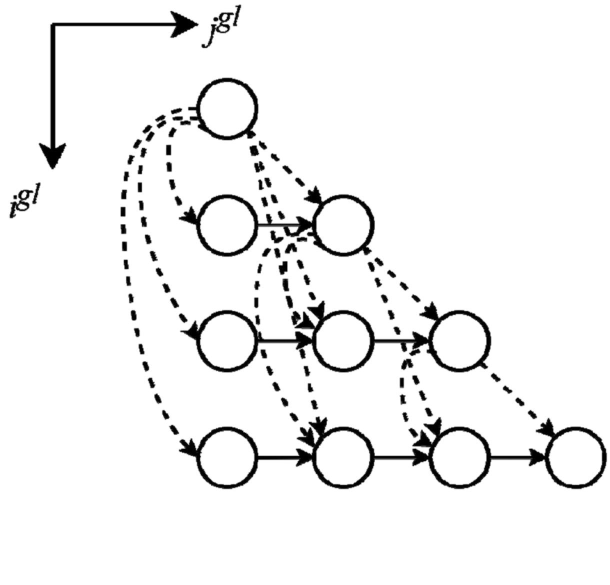

Рис. 2. Граф глобальных зависимостей, порождаемых зависимостями между операциями $S_3(i,j)$ $(Q_1=Q_2=4)$

Fig. 2. Graph presents the tiles dependences. The dependences in the statement of $S_3(i,j)$ are shown $(Q_1=Q_2=4)$

Найдем множества информационно зависимых операций тайлов. Из представления (4) имеем

$$V^{3,in_1}_{\left(J_1^{gl},J_2^{gl}\right),3,3}=\left\{\left(J_1,J_2\right)\in V^3_{\left(J_1^{gl},J_2^{gl}\right)}\,\middle|\,\begin{pmatrix}0&1\\0&1\end{pmatrix}\begin{pmatrix}J_1\\J_2\end{pmatrix}\in V^3_{\left(I_1^{gl},I_2^{gl}\right)},\ \left(I_1^{gl},I_2^{gl}\right)=\begin{pmatrix}0&1\\0&1\end{pmatrix}\begin{pmatrix}J_1^{gl}\\J_2^{gl}\end{pmatrix}\right\}=$$

$$=\left\{\left(J_1,J_2\right)\in Z^2\,\middle|\,2+J_1^{gl}r\le J_1\le 1+\left(J_1^{gl}+1\right)r,\ 1+J_2^{gl}r\le J_2\le\left(J_2^{gl}+1\right)r,\right.$$

$$\left.2+I_1^{gl}r\le J_2\le 1+\left(I_1^{gl}+1\right)r,\ 1+I_2^{gl}r\le J_2-1\le\left(I_2^{gl}+1\right)r,\ I_1^{gl}=J_2^{gl},\ I_2^{gl}=J_2^{gl}\right\}=$$

$$=\left\{\left(J_1,J_2\right)\in Z^2\,\middle|\,2+J_1^{gl}r\le J_1\le 1+\left(J_1^{gl}+1\right)r,\ 2+J_2^{gl}r\le J_2\le\left(J_2^{gl}+1\right)r\right\}.$$

Аналогично

$$V^{3,in_2}_{\left(J_1^{gl},J_2^{gl}\right),3,3}=\left\{\left(J_1,J_2\right)\in Z^2\,\middle|\,2+J_1^{gl}r\le J_1\le 1+\left(J_1^{gl}+1\right)r,\ J_2=1+J_2^{gl}r\right\}.$$

Из представления (5) получим

$$V^{3,out_1}_{\left(I_1^{gl},I_2^{gl}\right),3,3}=\left\{\left(I_1,I_2\right)\in Z^2\,\middle|\,2+I_1^{gl}r\le I_1\le\left(I_1^{gl}+1\right)r,\ I_2=I_1-1\right\},\ I_1^{gl}=I_2^{gl},$$

$$V^{3,out_2}_{\left(I_1^{gl},I_2^{gl}\right),3,3}=\left\{\left(I_1,I_2\right)\in Z^2\,\middle|\,I_1=1+\left(I_1^{gl}+1\right)r,\ I_2=I_1-1\right\},\ I_1^{gl}=I_2^{gl}.$$

На рис. 3 изображены очертания тайлов операций $S_3(i,j)$, отмечены множества зависимых операций тайлов и соответствующие векторы глобальных зависимостей.

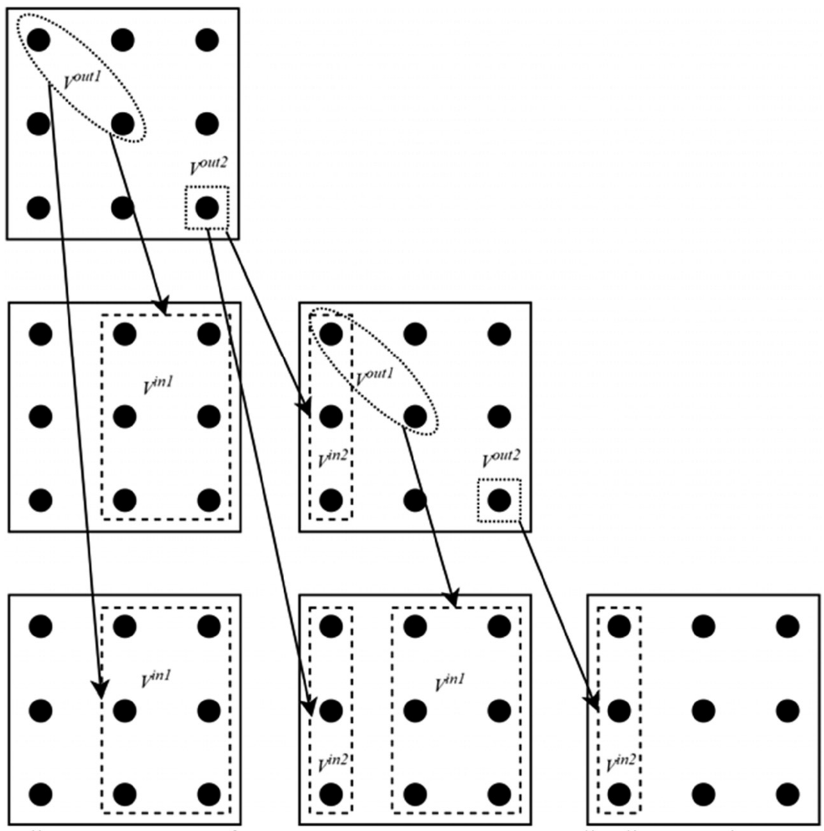

Рис. 3. Схема тайлов, множеств информационно зависимых операций тайлов, глобальных зависимостей ($r_1 = r_2 = 3$, $Q_1 = Q_2 = 3$)
Fig. 3. Tiles, communication sets of the tiles, global dependences ($r_1 = r_2 = 3$, $Q_1 = Q_2 = 3$)

3. Формализация коммуникационных операций. Перейдем непосредственно к формализации коммуникационных операций получения и отправки массивов данных в параллельном зернистом алгоритме с аффинными зависимостями, который предназначен для реализации на компьютере с распределенной памятью. Организуем зернистые вычислительные процессы (отобразим тайлы на 1D [14] или 2D вычислительные процессы) следующим образом: к одному процессу отнесем вычисления тайлов J^{gl} с одинаковыми значениями некоторых функций $\mathrm{Pr}^{\theta}(J^{gl})$, $1 \le \theta \le \Theta$.

Обозначим через $\overline{F}_{a,S_\beta,q}\left(V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}\right)$ образ множества $V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}$ после применения отображения $\overline{F}_{a,S_\beta,q}$. Через $\left(\overline{\Phi}^{gl_m}_{\alpha,\beta}\right)^{-1}\left(I^{gl}\right)$ обозначим множество $\left\{J^{gl} \mid \overline{\Phi}^{gl_m}_{\alpha,\beta}\left(J^{gl}\right) = I^{gl},\ J^{gl} \in V^{gl_m}_{\alpha,\beta}\right\}$. Функция (вообще говоря, многозначная) $\left(\overline{\Phi}^{gl_m}_{\alpha,\beta}\right)^{-1}$ задает итерации, на которых потребуются данные, вычисленные на итерации I^{gl}.

Обмены данными порождаются истинными зависимостями между операциями алгоритма. Информационно зависимые операции разных тайлов порождают глобальные зависимости между макрооперациями-тайлами зернистого алгоритма. Выше были формализованы множества информационно зависимых операций тайлов и глобальные зависимости между тайлами. Этой информации, если иметь еще и информацию о распределении тайлов между вычислительными процессами, достаточно для формализованного получения коммуникационных операций зернистого алгоритма.

Таким образом, можно сформулировать следующий результат работы.

У т в е р ж д е н и е. *Пусть зафиксированы функция глобальных зависимостей* $\overline{\Phi}^{gl_m}_{\alpha,\beta}$ *и тайл* J^{gl}. *Если* $\mathrm{Pr}^{\theta^\beta}\left(J^{gl}\right) \ne \mathrm{Pr}^{\theta^\alpha}\left(\overline{\Phi}^{gl_m}_{\alpha,\beta}\left(J^{gl}\right)\right)$, *то в алгоритме, задающем параллельные зернистые вычислительные процессы, возникает коммуникационная операция получения процессом* $\mathrm{Pr}^{\theta^\beta}\left(J^{gl}\right)$ *массива данных:*

$$\mathrm{receive}\left(\mathrm{Pr}^{\theta^\alpha}\left(\overline{\Phi}^{gl_m}_{\alpha,\beta}\left(J^{gl}\right)\right);\ a(F),\ F \in \overline{F}_{a,S_\beta,q}\left(V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}\right);\ \left|\overline{F}_{a,S_\beta,q}\left(V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}\right)\right|\right), \tag{7}$$

где первый аргумент обозначает процесс, в котором вычислялся массив, второй – пересылаемый массив, третий – указывает объем (число элементов) массива. Операция (7) *записывается перед записью тайла* J^{gl}.

Пусть зафиксирован тайл I^{gl}. *Если* $\left\{\mathrm{Pr}^{\theta^\beta}\left(\left(\overline{\Phi}^{gl_m}_{\alpha,\beta}\right)^{-1}\left(I^{gl}\right)\right)\right\} = \left\{\mathrm{Pr}^{\theta^\alpha}\left(I^{gl}\right)\right\}$, *то в алгоритме, задающем параллельные зернистые вычислительные процессы, возникает коммуникационная операция отправки процессом* $\mathrm{Pr}^{\theta^\alpha}\left(I^{gl}\right)$ *массива данных:*

$$\mathrm{send}\left(\left\{\mathrm{Pr}^{\theta^\beta}\left(\left(\overline{\Phi}^{gl_m}_{\alpha,\beta}\right)^{-1}\left(I^{gl}\right)\right)\right\};\ a\left(\overline{F}_{a,S_\alpha,1}(I)\right),\ I \in V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta};\ \left|V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}\right|\right), \tag{8}$$

где первый аргумент обозначает процесс (процессы), которому (которым) потребуются вычисленные элементы массива, второй – пересылаемый массив, третий – указывает объем массива. Операция (8) *записывается после записи тайла* I^{gl}.

Заметим, что для формализации коммуникационных операций можно не находить множества $V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}$ и $V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}$, нужны множества $\overline{F}_{a,S_\beta,q}\left(V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}\right)$ и $\overline{F}_{a,S_\alpha,1}\left(V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}\right)$; $\left|V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}\right| = \left|\overline{F}_{a,S_\alpha,1}\left(V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}\right)\right|$.

Заметим также, что в настоящей работе исследуется задача получения коммуникационных операций, но не задача уменьшения числа и объема коммуникационных операций [12, 15–17].

П р и м е р (продолжение). Продемонстрируем генерацию коммуникационных операций, порождаемых аффинной зависимостью между операциями $S_3(i,j)$. Пусть в рассмотренном в разделе 1 зернистом алгоритме одна итерация цикла do $i^{gl} = 0,\ Q_1 - 1$ выполняется одним процессом. Это означает, что по определению $\mathrm{Pr}^{S_3}(i^{gl}, j^{gl}) = i^{gl}$. Так как, с учетом вида (6) функций глобальных зависимостей и областей определения функций $\overline{\Phi}_{3,3}^{gl_m}$,

$$\mathrm{Pr}^{S_3}\left(\overline{\Phi}_{3,3}^{gl_1}\left(i^{gl}, j^{gl}\right)\right) = \mathrm{Pr}^{S_3}\left(\begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} i^{gl} \\ j^{gl} \end{pmatrix}\right) = j^{gl},\ j^{gl} < i^{gl},$$

$$\mathrm{Pr}^{S_3}\left(\overline{\Phi}_{3,3}^{gl_2}\left(i^{gl}, j^{gl}\right)\right) = \mathrm{Pr}^{S_3}\left(\begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} i^{gl} \\ j^{gl} \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix}\right) = j^{gl} - 1,\ j^{gl} \le i^{gl},\ j^{gl} \ne 1,$$

то глобальные зависимости, задаваемые функциями (6), порождают коммуникационные операции.

Формализуем, используя предлагаемый подход, эти коммуникационные операции (случай $r_1 = r_2 = r$). Так как с учетом вида множества $V^{3,in_1}_{\left(J_1^{gl}, J_2^{gl}\right),3,3}$, полученного в разделе 2,

$$\overline{F}_{x,S_3,3}\left(V^{3,in_1}_{\left(J_1^{gl}, J_2^{gl}\right),3,3}\right) = \left\{J_2 \mid\ 2 + J_2^{gl} r \le J_2 \le \left(J_2^{gl} + 1\right) r\right\},$$

то коммуникационная операция получения массива данных (7) принимает вид

$$\mathrm{receive1}\left\{j^{gl},\ j^{gl} < i^{gl};\ x\left(j\right), 2 + j^{gl} r \le j \le \left(j^{gl} + 1\right) r;\ r - 1\right\}.$$

Аналогично, так как

$$\overline{F}_{x,S_3,3}\left(V^{3,in_2}_{\left(J_1^{gl}, J_2^{gl}\right),3,3}\right) = \left\{J_2 \mid\ J_2 = 1 + J_2^{gl} r\right\},$$

то приходим к коммуникационной операции

$$\mathrm{receive2}\left\{j^{gl} - 1, j^{gl} \le i^{gl}, j^{gl} \ne 1;\ x\left(j\right), j = 1 + j^{gl} r;\ 1\right\}.$$

Получим коммуникационные операции отправки данных. Так как

$$\left(\overline{\Phi}_{3,3}^{gl_m}\right)^{-1}\left(I^{gl}(i^{gl}, j^{gl})\right) = \left\{(k, i^{gl}) \in Z^2 \mid i^{gl} + 1 \le k \le Q_1 - 1\right\}\ [4],$$

$$\overline{F}_{x,S_3,1}\left(V^{3,out_1}_{\left(I_1^{gl}, I_2^{gl}\right),3,3}\right) = \left\{I_1 \mid 2 + I_1^{gl} r \le I_1 \le \left(I_1^{gl} + 1\right) r\right\},\ I_1^{gl} = I_2^{gl},$$

$$\overline{F}_{x,S_3,1}\left(V^{3,out_2}_{\left(I_1^{gl}, I_2^{gl}\right),3,3}\right) = \left\{I_1 \mid I_1 = 1 + \left(I_1^{gl} + 1\right) r\right\},\ I_1^{gl} = I_2^{gl},$$

то коммуникационные операции отправки данных (8) принимают вид

$$\mathrm{send1}\left\{k, i^{gl} + 1 \le k \le Q_1 - 1;\ x(i),\ 2 + i^{gl} r \le i \le (i^{gl} + 1) r;\ r - 1\right\},\ i^{gl} = j^{gl},$$

$$\mathrm{send2}\left\{k, i^{gl} + 1 \le k \le Q_1 - 1;\ x(i),\ i = 1 + (i^{gl} + 1) r;\ 1\right\},\ i^{gl} = j^{gl}.$$

Структура кода для каждого из Q_1 процессов с номером $p (p = i^{gl})$, включающего найденные коммуникационные операции, имеет следующий вид:

```
if p = 0   S1: x(1) = b(1)
    do i = 2 + p r, min(1 + (p + 1)r, n) // Цикл с параметром i задает тайл Tile1(p)
        S2: x(i) = b(i)
    enddo
    do j^gl = 0, Q2 – 1 // Одна итерация цикла – «обработка» каждым
                          процессом «своих» r столбцов A
      if p > 0 // Получение данных для выполнения S3 на вхождении x(j)
        receive1{j^gl, j^gl < p; x(j), 2 + j^gl r ≤ j ≤ (j^gl + 1)r; r – 1}
        receive2{j^gl – 1, j^gl ≤ p, j^gl ≠ 1; x(j), j = 1 + j^gl r; 1}
      endif
        do i = 2 + p r, min(1 + (i^gl + 1)_1, n) // Циклы с параметрами i и j задают Tile2(p,j^gl)
            do j = 1 + j^gl r, min((j^gl + 1)r, i – 1)
                S3: x(i) = x(i) – a(i,j)x(j)
            enddo
        enddo
      if p = j^gl и p < Q1 – 1 // Отправка части данных,
                                вычисленных на вхождении x(i)
        send1{k, p + 1 ≤ k ≤ Q1 – 1; x(i), 2 + p r ≤ i ≤ (p + 1)r; r – 1}
        send2{k, p + 1 ≤ k ≤ Q1 – 1; x(i), i = 1 + (p + 1)r; 1}
      endif
    enddo
```

Заключение. Таким образом, для случая алгоритмов с аффинными зависимостями представлен метод формализации и включения в схему алгоритма коммуникационных операций уровня тайлов (т. е. уровня макроопераций). Данный подход использует структуру информационных зависимостей исходного алгоритма уровня макроопераций, обобщает метод формализации обменных операций при отображении на компьютеры с распределенной памятью алгоритмов с однородными зависимостями.

Укажем возможные направления дальнейших исследований.

1. Нахождение коммуникационных операций с учетом возможного дублирования элементов массивов, требуемых на разных вхождениях в оператор (операторы) одного массива данных (одновременно получаемые процессом множества $a\left(\overline{F}_{a,S_\beta,q}\left(J\right)\right)$, $J \in V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}$ могут пересекаться).

2. Исследование возможности аналитического представления множеств $V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}$ и $V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}$, $\overline{F}_{a,S_\beta,q}\left(V^{\theta^\beta,in_m}_{J^{gl},\alpha,\beta}\right)$ и $\overline{F}_{a,S_\alpha,1}\left(V^{\theta^\alpha,out_m}_{I^{gl},\alpha,\beta}\right)$.

3. Использование предлагаемого метода для автоматизации генерации коммуникационных операций. Сравнение с генерацией коммуникационных операций системой PLUTO [9].

4. Разработка параллельных алгоритмов для прикладных задач с использованием предлагаемого подхода к получению коммуникационных операций.

Благодарности. Работа выполнена в рамках Государственной программы научных исследований Республики Беларусь «Конвергенция-2020», подпрограмма «Методы математического моделирования сложных систем».

Acknowledgments. The work was sponsored by the Government Research Program of the Republic of Belarus "Convergence-2020", Subprogram "Methods of Mathematical Modeling of Complex Systems".

Список использованных источников

1. Xue, J. Time-minimal tiling when rise is larger than zero / J. Xue, W. Cai // Parallel Computing. – 2002. – Vol. 28, №. 5. – P. 915–939.

2. Kim, D. Parameterized tiling for imperfectly nested loops: Technical Report CS-09-101 / D. Kim, S. Rajopadhye. – Colorado State University, Department of Computer Science, February 2009. – 21 p.

3. Dathathri, R. Compiling Affine Loop Nests for a Dynamic Scheduling Runtime on Shared and Distributed Memory / R. Dathathri, R. T. Mullapudi, U. Bondhugula // ACM Transactions on Parallel Computing (TOPC). – 2016. – Vol. 3, №. 2. – P. 1–28.

4. Лиходед, Н. А. Формализация коммуникационных операций многомерных циклов / Н. А. Лиходед, А. А. Толстиков // Вес. Нац. акад. навук Беларусі. Сер. фіз.-мат. навук. – 2010. – № 3. – С. 109–114.

5. Лиходед, Н. А. Коммуникационные операции параллельного алгоритма, порождаемые однородными зависимостями / Н. А. Лиходед, П. И. Соболевский, А. А. Толстиков // Докл. Нац. акад. наук Беларуси. – 2011. – Т. 55, № 3. – С. 21–26.

6. Лиходед, Н. А. Информационная структура зернистых алгоритмов с однородными зависимостями / Н. А. Лиходед, П. И. Соболевский // Докл. Нац. акад. наук Беларуси. – 2011. – Т. 55, № 2. – С. 22–26.

7. Bondhugula, U. Compiling affine loop nests for distributed-memory parallel architectures / U. Bondhugula // Supercomputing 2013: Proc. of the Int. Conf. on High Performance Computing, Networking, Storage and Analysis. ACM, 2013. – Article №. 33.

8. Generating efficient data movement code for heterogeneous architectures with distributed-memory / R. Dathathri [et al.] // Proc. of the 22th Int. Conf. on Parallel Architectures and Compilation Techniques (PACT), 2013. – P. 375–386.

9. PLUTO: An automatic parallelizer and locality optimizer for affine loop nests [Electronic resource]. – Mode of access: http://pluto-compiler.sourceforge.net. – Date of access: 23.10.2017.

10. Воеводин, В. В. Вычислительная математика и структура алгоритмов / В. В. Воеводин. – М.: Изд-во МГУ, 2006. – 112 с.

11. Лиходед, Н. А. Функции, задающие зависимости зернистых алгоритмов / Н. А. Лиходед, А. А. Толстиков // Докл. Нац. акад. наук Беларуси. – 2014. – Т. 58, № 4. – С. 35–41.

12. Воеводин, В. В. Параллельные вычисления / В. В. Воеводин, Вл. В. Воеводин. – СПб.: БХВ-Петербург, 2002. – 608 с.

13. Воеводин, В. В. Массивный параллелизм и декомпозиция алгоритмов / В. В. Воеводин // Журн. вычисл. математики и мат. физики. – 1995. – Т. 35, № 6. – С. 988–996.

14. Лиходед, Н. А. Параллельные последовательности зернистых вычислений / Н. А. Лиходед, А. А. Толстиков // Докл. Нац. акад. наук Беларуси. – 2010. – Т. 54, № 4. – С. 36–41.

15. Lim, A. W. Maximizing parallelism and minimizing synchronization with affine partitions / A. W. Lim, M. S. Lam // Parallel Computing. – 1998. – Vol. 24, № 3/4. –P. 445–475.

16. Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model / U. Bondhugula [et al.] // Lecture Notes in Computer Science. – 2008. – № 4959. – P. 132–146.

17. Лиходед, Н. А. Достаточные условия определения и использования данных в одном параллельном зернистом вычислительном процессе / Н. А. Лиходед // Журн. вычисл. математики и мат. физики. – 2014. – Т. 54, № 8. – С. 1356–1367.

References

1. Xue J., Cai W. Time-minimal tiling when rise is larger than zero. *Parallel Computing*, 2002, vol. 28, no. 5, pp. 915–939. Doi: 10.1016/s0167-8191(02)00098-4

2. Kim D., Rajopadhye S. *Parameterized Tiling for Imperfectly Nested Loops. Technical Report CS-09-101.* Colorado State University, Department of Computer Science, February 2009. 21 p.

3. 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, pp. 1–28. Doi: 10.1145/2948975

4. Likhoded N. A., Tolstikov A. A. Formalization of communication operations of multidimensional loops. *Vestsi Natsyianal'nai akademii navuk Belarusi. Seryia fizika-matematychnykh navuk = Proceedings of the National Academy of Sciences of Belarus. Physics and Mathematics series*, 2010, no. 3, pp. 109–114 (in Russian).

5. Likhoded N. A., Sobolevsky P. I., Tolstikov A. A. Parallel algorithm communication operations generated by uniform dependences. *Doklady Natsional'noi akademii nauk Belarusi = Doklady of the National Academy of Sciences of Belarus*, 2011, vol. 55, no. 3, pp. 21–26 (in Russian).

6. Likhoded N. A., Sobolevsky P. I. Information structure of grained algorithms with homogeneous dependencies. *Doklady Natsional'noi akademii nauk Belarusi = Doklady of the National Academy of Sciences of Belarus*, 2011, vol. 55, no. 2, pp. 22–26 (in Russian).

7. Bondhugula U. Compiling affine loop nests for distributed-memory parallel architectures. *Supercomputing 2013. Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, ACM, 2013.* Article №. 33. Doi: 10.1145/2503210.2503289

8. Dathathri R., Reddy C., Ramashekar T., Bondhugula U. Generating efficient data movement code for heterogeneous architectures with distributed-memory. *Proceedings of the 22th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2013, pp. 375–386. Doi: 10.1109/pact.2013.6618833

9. *PLUTO: An automatic parallelizer and locality optimizer for affine loop nests*. Available at: http://pluto-compiler.sourceforge.net (accessed 23 October 2017).

10. Voevodin V. V. *Computational mathematics and the structure of algorithms*. Moscow, Moscow State University Publishing House, 2006. 112 p. (in Russian).

11. Likhoded N. A., Tolstikov A. A. Functions assigning the dependences of grained algorithms. *Doklady Natsional'noi akademii nauk Belarusi = Doklady of the National Academy of Sciences of Belarus*, 2014, vol. 58, no. 4, pp. 35–41 (in Russian).

12. Voevodin V. V., Voevodin Vl. V. *Parallel Computations*. Saint Petersburg, BKhV-Peterburg Publ., 2002. 608 p. (in Russian).

13. Voevodin V. V. Massive parallelism and decomposition of algorithms. *Zhurnal vychislitel'noi matematiki i matematicheskoi fiziki = Computational Mathematics and Mathematical Physics*, 1995, vol. 35, no. 6, pp. 988–996 (in Russian).

14. Likhoded N. A., Tolstikov A. A. Parallel sequences of grain computations. *Doklady Natsional'noi akademii nauk Belarusi = Doklady of the National Academy of Sciences of Belarus*, 2010, vol. 54, no. 4, pp. 36–41 (in Russian).

15. Lim A. W., Lam M. S. Maximizing parallelism and minimizing synchronization with affine partitions. *Parallel Computing*, 1998, vol. 24, no. 3–4, pp. 445–475. Doi: 10.1016/s0167-8191(98)00021-0

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*, 2008, no. 4959, pp. 132–146. Doi: 10.1007/978-3-540-78791-4_9

17. 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, pp. 1316–1326. Doi: 10.1134/s0965542514080077

Информация об авторах

Лиходед Николай Александрович – доктор физико-математических наук, профессор кафедры вычислительной математики факультета прикладной математики и информатики, Белорусский государственный университет (пр. Независимости, 4, 220030, г. Минск, Республика Беларусь), E-mail:likhoded@bsu.by

Толстиков Алексей Александрович – старший преподаватель кафедры вычислительной математики факультета прикладной математики и информатики, Белорусский государственный университет (пр. Независимости, 4, 220030, г. Минск, Республика Беларусь), E-mail: tolstikov @bsu.by

Information about the authors

Nikolai A. Likhoded – D. Sc. (Physics and Mathematics), Professor, Belarusian State University (4, Nezavisimosti Str., 220030, Minsk, Republic of Belarus). E-mail:likhoded@bsu.by

Aliaksei A. Tolstsikau – Senior Lecturer, Belarusian State University (4, Nezavisimosti Str., 220030, Minsk, Republic of Belarus). E-mail: tolstikov@bsu.by