ПРОГРАММНО-ТЕХНИЧЕСКИЕ КОМПЛЕКСЫ

Е.В. АДУЦКЕВИЧ, Н.А. ЛИХОДЕД, А.О. СИКОРСКИЙ

УДК 519.6+681.3.012

К РАСПАРАЛЛЕЛИВАНИЮ ПОСЛЕДОВАТЕЛЬНЫХ ПРОГРАММ: РАСПРЕДЕЛЕНИЕ МАССИВОВ МЕЖДУ ПРОЦЕССОРАМИ И СТРУКТУРИЗАЦИЯ КОММУНИКАЦИЙ

Ключевые слова: *параллельные вычисления, распараллеливание алгоритмов, аффинные гнезда циклов, параллельный компьютер с распределенной памятью, оптимизация коммуникаций.*

ВВЕДЕНИЕ

При адаптации последовательных алгоритмов для реализации на параллельных компьютерах с распределенной памятью необходимо решить две основные задачи.

Первая задача заключается в распределении операций алгоритма между процессорами и установлении нового порядка их выполнения. Основная проблема, возникающая при решении этой задачи, состоит в сохранении порядка выполнения информационно-связанных операций — выполнении условий сохранения зависимостей алгоритма. Исследованию этой задачи посвящено много публикаций (в частности, [1–5]), в которых найдены различные способы ее решения.

Вторая задача заключается в распределении данных алгоритма между процессорами и установлении схемы обмена данными при выполнении параллельного алгоритма. Основная проблема, возникающая при решении задачи, состоит в необходимости устанавливать в определенный момент времени местоположение требуемого для выполнения операции данного и дополнять вычислительный алгоритм новыми операциями передачи и приема данных. При этом следует учитывать, что реализация коммуникаций на параллельном компьютере с распределенной памятью требует значительных временных затрат. Поскольку целью использования параллельных компьютеров является уменьшение времени решения задачи, то при распараллеливании алгоритма необходимо стремиться к уменьшению коммуникационных затрат на его реализацию.

В отличие от первой задачи вторая задача является менее изученной. Сделаем краткий обзор подходов, применяемых к решению этой задачи.

Один из очевидных подходов к распределению данных и минимизации коммуникационных затрат заключается в разбиении алгоритма на блоки независимых вычислений [1, 6–9]. В этом случае распределение данных осуществляется в соответствии с распределением операций, которые эти данные используют, поэтому все процессоры параллельной вычислительной системы могут работать независимо один от другого, не нуждаясь в обмене данными. Очевидно, такой

подход упрощает проблему распределения данных между процессорами и устраняет проблему обмена данными, однако на практике алгоритм редко допускает декомпозицию на независимые части.

Другой подход заключается в получении блочных версий алгоритма, т.е. разбиении специальным образом пространства итераций гнезд циклов [1, 10–13]. Целью такого разбиения является увеличение пакета передаваемых данных и уменьшение частоты коммуникаций. Подход направлен на минимизацию накладных расходов на обмены данными, не затрагивая проблемы размещения массивов данных.

Существует также подход, заключающийся в установлении фиксированного (определенного до начала выполнения программы и не меняющегося в процессе ее выполнения) размещения данных по процессорам вычислительной системы [4, 14–19]. Поиск такого размещения данных осуществляется, исходя из условия минимизации обменов данными в процессе выполнения параллельного алгоритма между процессорами, в локальной памяти которых хранятся данные, и процессорами, в которых эти данные вычисляются или используются как аргументы для вычислений. В рамках подхода в работах [4, 16, 17] предлагаются алгоритмы совместного распределения операций и данных алгоритма по процессорам. В работах [18, 19] рассматриваются условия, при выполнении которых возможна организация структурированных коммуникаций, которые реализуются между группой процессоров и могут выполняться быстрее, чем соответствующий набор коммуникаций между парами процессоров. К преимуществам подхода можно отнести упрощение схемы обмена данными (обмен происходит всегда от процессора, который использует данное в качестве результата вычислений, к процессору, в локальной памяти которого данное хранится, и обратно — от одного процессора, в локальной памяти которого хранится данное, к другому процессору, который использует данное как аргумент для вычислений). К преимуществам подхода относится также возможность совместного нахождения распределения операций и данных, направленного на уменьшение коммуникаций, требуемых в процессе реализации алгоритма.

Однако для большого числа вычислительных алгоритмов характерна ситуация, когда элементы массивов переопределяются на протяжении всего времени выполнения алгоритма. В этом случае применение такого подхода приведет к организации большого числа избыточных коммуникаций между процессорами, использующими элемент массива для вычислений, и процессором, назначенным для его хранения, вместо того, чтобы осуществлять непосредственную передачу данного между процессорами, использующими элемент массива по очереди. Для выявления такой ситуации и изменения схемы обмена данными в рамках рассматриваемого подхода необходимо проводить дополнительные исследования алгоритма, как это показано в работе [18]. Другие решения проблемы предложены в [20, 21].

В настоящей статье рассмотрен новый подход к поиску распределения данных алгоритма между процессорами и итерациями параллельного алгоритма. При этом с учетом, что распределение операций алгоритма между процессорами и итерациями параллельного алгоритма уже найдено, устанавливаются процессоры и итерации, использующие при выполнении программы элемент массива на фиксированном его вхождении в оператор. Данная информация позволяет получать начальное распределение массивов (в том числе приводящее к использованию каждого элемента массива только одним процессором), информацию об объеме массивов, используемых в каждом процессоре (что необходимо для резервирования памяти процессора при записи программы), устанавливать коммуникации, требуемые при заданном распределении операций, и определять возможность структуризации коммуникаций. Полученные результаты пригодны для

автоматизации, что делает возможным их использование при разработке программных комплексов, предназначенных для распараллеливания последовательных алгоритмов.

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

Пусть алгоритм задан аффинным гнездом циклов: индексные выражения переменных и границы изменения параметров циклов алгоритма являются аффинными функциями от параметров циклов и внешних переменных.

Пусть в гнезде циклов имеется K выполняемых операторов S_β и используется L массивов a_l. Простые переменные будем считать массивами размерности 0; $V_\beta \subset \mathbf{Z}^{n_\beta}$ — область изменения параметров гнезда циклов для оператора S_β, где n_β — число циклов, окружающих оператор S_β; $W_l \subset \mathbf{Z}^{\nu_l}$ — область изменения индексов l-го массива, где ν_l — размерность l-го массива; $N \in \mathbf{Z}^e$ — вектор внешних переменных алгоритма, где e — число внешних переменных. Реализацию оператора S_β при конкретном значении вектора параметров циклов J будем называть операцией и обозначать $S_\beta(J)$. Выполнение всех операций, зависящих от J, называется J-й итерацией.

Вхождением (l, β, q) будем называть q-е вхождение массива a_l в оператор S_β. Другими словами, вхождение (l, β, q) — это q-е обращение в последовательности обращений к элементам массива a_l при очередном выполнении оператора S_β. Для наглядности наряду с (l, β, q) будем использовать также обозначение (a_l, S_β, q). Область изменения индексов элементов массива $a_l(F)$, связанных с вхождением $(l, \beta\, q)$, будем обозначать $W_{l,\beta,q}$. Обозначим Q множество вхождений (l, β, q) всех массивов a_l во все операторы S_β (берется объединение вхождений (l, β, q) по всем l, β и q).

Индексы элементов l-го массива, связанных с вхождением (l, β, q), выражаются аффинной функцией $\overline{F}_{l,\beta,q}: V_\beta \to W_l$ вида

$$\overline{F}_{l,\beta,q}(J) = F_{l,\beta,q} J + G_{l,\beta,q} N + f^{(l,\beta,q)},$$
$$J \in V_\beta,\ N \in \mathbf{Z}^e,\ F_{l,\beta,q} \in \mathbf{Z}^{\nu_l \times n_\beta},\ G_{l,\beta,q} \in \mathbf{Z}^{\nu_l \times e},\ f^{(l,\beta,q)} \in \mathbf{Z}^{\nu_l},\ (l,\beta,q) \in Q. \tag{1}$$

Пример 1. Пусть A — левая треугольная матрица порядка N с диагональными элементами, равными единице. Рассмотрим алгоритм решения системы линейных алгебраических уравнений $Ax = b$ методом обратной подстановки:

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

Гнездо циклов содержит три оператора: S_1, S_2, S_3 и элементы трех массивов: x, b, x. В данном случае $n_1 = 0$, $n_2 = 1$, $n_3 = 2$, $\nu_1 = 1$, $\nu_2 = 1$, $\nu_3 = 2$, $V_1 = \{(1)\}$, $V_2 = \{(i) \in \mathbf{Z} \mid 2 \le i \le N\}$, $V_3 = \{(i, j) \in \mathbf{Z}^2 \mid 2 \le i \le N, 1 \le j \le i-1\}$, $W_1 = W_2 = \{(i) \in \mathbf{Z} \mid 1 \le i \le N\}$, $W_3 = V_3$.

Индексы элементов массивов, используемых операторами алгоритма, выражаются функциями $\overline{F}_{x,S_1,1}(1) = 1$, $\overline{F}_{x,S_2,1}(i) = E^{(1)}(i)$, $\overline{F}_{x,S_3,1}(i, j) = \overline{F}_{x,S_3,2}(i, j) =$

$= (1\ \ 0)(i\ j)^{\mathrm{T}}$, $\overline{F}_{x,S_3,3}(i,j) = (0\ \ 1)(i\ j)^{\mathrm{T}}$, $\overline{F}_{b,S_1,1}(1) = 1$, $\overline{F}_{b,S_2,1}(i) = E^{(1)}(i)$, $\overline{F}_{a,S_3,1}(i,j) =$
$= E^{(2)}(i\ j)^{\mathrm{T}}$. ■

Операция $S_\beta(J)$, $J \in V_\beta$, зависит от операции $S_\alpha(I)$, $I \in V_\alpha$, если [22]:

1) $S_\alpha(I)$ выполняется раньше, чем $S_\beta(J)$;

2) $S_\alpha(I)$ и $S_\beta(J)$ используют один и тот же элемент массива (т.е. $\overline{F}_{l,\alpha,p}(I) = \overline{F}_{l,\beta,q}(J)$ для некоторых l, p, q) и по крайней мере одно из использований представляет переопределение (изменение) элемента;

3) между операциями $S_\alpha(I)$ и $S_\beta(J)$ этот элемент не переопределяется.

Зависимость может быть истинной (если элемент массива сначала определяется, а затем его значение используется в качестве аргумента), антизависимостью (если значение элемента массива сначала используется как аргумент, а затем переопределяется), зависимостью по выходу (если элемент массива определяется, а затем переопределяется).

Зависимость операции $S_\beta(J)$ от операции $S_\alpha(I)$ будем обозначать $S_\alpha(I) \to S_\beta(J)$. Пусть $P = \{(\alpha,\beta) \mid \exists\, I \in V_\alpha, J \in V_\beta, S_\alpha(I) \to S_\beta(J)\}$ — множество, которое определяет пары зависимых операторов. Для каждой пары $(\alpha,\beta) \in P$ обозначим $V_{\alpha,\beta} = \{J \in V_\beta \mid \exists\, S_\alpha(I) \to S_\beta(J)\}$. Отметим, что между операциями $S_\alpha(I)$ и $S_\beta(J)$ может быть более одной зависимости, если равенство $\overline{F}_{l,\alpha,p}(I) = \overline{F}_{l,\beta,q}(J)$ выполняется более чем для одного набора l, p, q. В этом случае можно использовать обозначение $S_\alpha(I) \overset{l,p,q}{\to} S_\beta(J)$.

Пусть зависимости операций алгоритма задаются функциями $\overline{\Phi}_{\alpha,\beta}: V_{\alpha,\beta} \to V_\alpha$ таким образом, что если $S_\alpha(I) \to S_\beta(J)$, $I \in V_\alpha$, $J \in V_{\alpha,\beta} \subseteq V_\beta$, то $I = \overline{\Phi}_{\alpha,\beta}(J)$. Если пара операторов S_α и S_β порождает более одной зависимости, то для обозначения функций $\overline{\Phi}_{\alpha,\beta}$ используются также верхние индексы. Будем называть функции $\overline{\Phi}_{\alpha,\beta}$ функциями зависимостей и считать, что они являются аффинными:

$$\overline{\Phi}_{\alpha,\beta}(J) = \Phi_{\alpha,\beta} J + \Psi_{\alpha,\beta} N - \varphi^{(\alpha,\beta)},$$
$$J \in V_{\alpha,\beta},\ N \in \mathbf{Z}^e,\ \Phi_{\alpha,\beta} \in \mathbf{Z}^{n_\alpha \times n_\beta},\ \Psi_{\alpha,\beta} \in \mathbf{Z}^{n_\alpha \times e},\ \varphi^{(\alpha,\beta)} \in \mathbf{Z}^{n_\alpha},\ (\alpha,\beta) \in P, \tag{2}$$

Пример 2 (продолжение примера 1). Зависимости гнезда циклов описываются следующими функциями [1, § 6.8]: $\overline{\Phi}_{1,3}(i,1) = (0\ \ 0)(i\ 1)^{\mathrm{T}} + 1$, $(i,1) \in V_{1,3} =$
$= \{(i,1) \in \mathbf{Z}^2 \mid 2 \le i \le N\}$, $\overline{\Phi}_{2,3}(i,1) = (1\ \ 0)(i\ 1)^{\mathrm{T}}$, $(i,1) \in V_{2,3} = V_{1,3}$, $\overline{\Phi}^{(1)}_{3,3}(i,j) =$
$= \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} (i\ j)^{\mathrm{T}} - (0\ \ 1)^{\mathrm{T}}$, $(i,j) \in V^{(1)}_{3,3} = \{(i,j) \in \mathbf{Z}^2 \mid 3 \le i \le N, 2 \le j \le i-1\}$, $\overline{\Phi}^{(2)}_{3,3}(i,j) =$
$= E^{(2)}(i\ j)^{\mathrm{T}} - (0\ \ 1)^{\mathrm{T}}$, $(i,j) \in V^{(2)}_{3,3} = V^{(1)}_{3,3}$. ■

Функции зависимостей являются удобным математическим аппаратом для описания тонкой информационной структуры алгоритма, т.е. для описания информационных связей на уровне отдельных операций алгоритма. Эти функции могут иметь другие названия: покрывающие функции графа алгоритма [1], *h*-преобразования [23, 24]. Доказано (теорема В.В. Воеводина об информационном покрытии [1, 25]), что все зависимости алгоритмов, представленных достаточно широким линейным классом программ, можно задать функциями, линейно зависящими от параметров циклов и внешних переменных. Каждая функция

определена на линейном многограннике; число функций не зависит от внешних переменных и для реальных линейных программ пропорционально числу операторов присваивания. Коэффициент пропорциональности не превосходит нескольких единиц.

Обозначим $n = \max\limits_{1\le\beta\le K} n_\beta$. Введем в рассмотрение векторные функции $\bar{t}^{(\beta)} : V_\beta \to \mathbf{Z}^n, 1 \le \beta \le K$. Пусть каждая функция $\bar{t}^{(\beta)}$ ставит в соответствие операции $S_\beta(J)$ вектор $\bar{t}^{(\beta)}(J) = (t_1^{(\beta)}(J), \dots, t_n^{(\beta)}(J))$ с целочисленными координатами. Будем предполагать, что функции $t_\xi^{(\beta)}$ являются аффинными:

$$t_\xi^{(\beta)}(J) = \tau^{(\beta,\xi)} J + b^{(\beta,\xi)} N + a_{\beta,\xi},$$
$$J \in V_\beta,\ \tau^{(\beta,\xi)} \in \mathbf{Z}^{n_\beta},\ b^{(\beta,\xi)},\ N \in \mathbf{Z}^e,\ a_{\beta,\xi} \in \mathbf{Z},\ 1 \le \beta \le K,\ 1 \le \xi \le n. \tag{3}$$

Функции $\bar{t}^{(\beta)}$ называются векторными таймирующими функциями, если выполняются следующие условия:

$$\operatorname{rank} T^{(\beta)} = n_\beta,\ 1 \le \beta \le K, \tag{4}$$

$$\bar{t}^{(\beta)}(J) \ge_{lex} \bar{t}^{(\alpha)}(I),\ J \in V_\beta,\ I \in V_\alpha,\ S_\alpha(I) \to S_\beta(J), \tag{5}$$

где $T^{(\beta)}$ — матрица, строки которой составлены из векторов $\tau^{(\beta,1)}, \dots, \tau^{(\beta,n)}$, запись $\ge_{lex}$ означает «лексикографически больше либо равно». Заметим, что в терминологии монографии [1] функции $t_\xi^{(\beta)}$ называются развертками графа алгоритма.

Векторные таймирующие функции задают преобразование гнезда циклов. Это значит, что операция $S_\beta(J)$, выполняемая на итерации J исходного гнезда циклов, будет выполняться на итерации $\bar{t}^{(\beta)}(J)$ преобразованного гнезда циклов. Таким образом, компоненты вектора $\bar{t}^{(\beta)}$ интерпретируются как параметры преобразованного гнезда циклов для оператора S_β : $t_1^{(\beta)}$ — параметр самого внешнего цикла, $t_n^{(\beta)}$ — параметр самого внутреннего цикла. Выполнение условия (4) гарантирует невырожденность преобразования, выполнение условия (5) гарантирует сохранение порядка выполнения информационно связанных операций алгоритма, что обеспечивает корректность преобразования. Набор векторных функций $\bar{t}^{(1)}, \dots, \bar{t}^{(K)}$ называется многомерным таймированием.

Пример 3 (продолжение примера 2). Рассмотрим двумерное таймирование: $\bar{t}^{(1)}(1) = (1,1)$; $\bar{t}^{(2)}(i) = (i,1),\ 2 \le i \le N$; $\bar{t}^{(3)}(i,j) = (i,j),\ 2 \le i \le N,\ 1 \le j \le i-1$. Преобразуем исходное гнездо циклов в соответствии с многомерным таймированием:

```
do i = 1, N
   do j = 1, i − 1
      if (i = j = 1)     S1 : x(1) = b(1)
      if (i ≥ 2, j = 1)  S2 : x(i) = b(i)
      if (i ≥ 2)         S3 : x(i) = x(i) − a(i, j)x(j)
   enddo
enddo
```

■

Обозначим $\rho_{l,\beta,q} = \operatorname{rank} F_{l,\beta,q}$. Рассмотрим базис пространства $\mathbf{Z}^{n_\beta}$ с базисными векторами $u_i^{\perp}, 1 \le i \le \rho_{l,\beta,q}$, $u_i, 1 \le i \le n_\beta - \rho_{l,\beta,q}$, где u_i — базисные векто-

ры подпространства $\ker F_{l,\beta,q}$; если $\rho_{l,\beta,q} = n_\beta$, то векторы u_i отсутствуют. Матрицу, столбцы которой составлены из векторов $u_i^{\perp}$, обозначим $U^{\perp}_{l,\beta,q}$.

Пусть F — элемент множества W_l. Обозначим $V_{l,\beta,q}(F)$ множество итераций исходного гнезда циклов, на вхождении (l, β, q) которых используется один и тот же элемент массива $a_l(F)$:

$$V_{l,\beta,q}(F) = \{ J \in V_\beta \mid \overline{F}_{l,\beta,q}(J) = F \}.$$

Обозначим $\Lambda_{l,\beta,q}(F)$ множество, задающее координаты проекции множества $V_{l,\beta,q}(F)$ на линейную оболочку векторов u_i:

$$\Lambda_{l,\beta,q}(F) = \left\{ (\lambda_1, \ldots, \lambda_{n_\beta - \rho_{l,\beta,q}}) \mid J = J^{\perp} + \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i u_i,\ J \in V_{l,\beta,q}(F) \right\};$$

в случае $\rho_{l,\beta,q} = n_\beta$ множество $\Lambda_{l,\beta,q}(F)$ пустое.

ФУНКЦИИ ИСПОЛЬЗОВАНИЯ МАССИВОВ НА ИТЕРАЦИЯХ ГНЕЗДА ЦИКЛОВ

Введем в рассмотрение функции $\overline{d}^{(l,\beta,q)} : W_{l,\beta,q} \to \mathbf{Z}^n$, $(l, \beta, q) \in Q$, с аффинными координатными функциями вида

$$\begin{gathered} d_\xi^{(l,\beta,q)}(F) = \eta^{(l,\beta,q,\xi)} F + z^{(l,\beta,q,\xi)} N + y_{(l,\beta,q,\xi)}(F), \\ F \in W_{l,\beta,q},\ \eta^{(l,\beta,q,\xi)} \in \mathbf{Z}^{\nu_l},\ z^{(l,\beta,q,\xi)},\ N \in \mathbf{Z}^e, \\ y_{l,\beta,q,\xi}(F) \in \mathbf{Z},\ (l, \beta, q) \in Q,\ 1 \le \xi \le n. \end{gathered} \tag{6}$$

Для каждого фиксированного $F \in W_{l,\beta,q}$ потребуем выполнения условия

$$d_\xi^{(l,\beta,q)}(F) = t_\xi^{(\beta)}(J),\ J \in V_{l,\beta,q}(F). \tag{7}$$

Тогда каждая функция $\overline{d}^{(l,\beta,q)}$ определяет итерации преобразованного гнезда циклов, на которых используются элементы массива a_l, связанного с вхождением (l, β, q). Будем называть функции $\overline{d}^{(l,\beta,q)}$ функциями использования массивов на итерациях гнезда циклов. Отметим, что функции $d_\xi^{(l,\beta,q)}$ в общем случае многозначные.

Теорема 1. Координаты $d_\xi^{(l,\beta,q)}$ функций использования массивов на итерациях гнезда циклов задаются формулами (6), в которых $\eta^{(l,\beta,q,\xi)}$ — решение системы уравнений

$$\eta^{(l,\beta,q,\xi)} F_{l,\beta,q} U^{\perp}_{l,\beta,q} = \tau^{(\beta,\xi)} U^{\perp}_{l,\beta,q}, \tag{8}$$

а векторы $z^{(l,\beta,q,\xi)}$ и величины $y_{l,\beta,q,\xi}(F)$ определяются равенствами

$$z^{(l,\beta,q,\xi)} = b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q}, \tag{9}$$

$$y_{l,\beta,q,\xi}(F) = a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} +$$

$$+ \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i,\ (\lambda_1, \ldots, \lambda_{n_\beta - \rho_{l,\beta,q}}) \in \Lambda_{l,\beta,q}(F). \tag{10}$$

Доказательство. Пусть F — произвольный фиксированный элемент множества $W_{l,\beta,q}$. Тогда $F = \overline{F}_{l,\beta,q}(J)$, $J \in V_{l,\beta,q}(F)$, и выполнение условия (7) рав-

носительно выполнению равенства $t_{\xi}^{(\beta)}(J) - d_{\xi}^{(l,\beta,q)}(\overline{F}_{l,\beta,q}(J)) = 0$ для любого $J \in V_{l,\beta,q}(F)$.

Имеем:

$$t_{\xi}^{(\beta)}(J) - d_{\xi}^{(l,\beta,q)}(\overline{F}_{l,\beta,q}(J)) = \tau^{(\beta,\xi)}J + b^{(\beta,\xi)}N + a_{\beta,\xi} - (\eta^{(l,\beta,q,\xi)}\overline{F}_{l,\beta,q}(J) +$$

$$+ z^{(l,\beta,q,\xi)}N + y_{l,\beta,q,\xi}) = \tau^{(\beta,\xi)}J + b^{(\beta,\xi)}N +$$

$$+ a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)}(F_{l,\beta,q}J + G_{l,\beta,q}N + f^{(l,\beta,q)}) - z^{(l,\beta,q,\xi)}N - y_{l,\beta,q,\xi} =$$

$$= (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q})J + (b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}G_{l,\beta,q} - z^{(l,\beta,q,\xi)})N +$$

$$+ a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)} - y_{l,\beta,q,\xi}.$$

Равенство (7) будет выполняться, если достичь равенства нулю каждой из величин $(\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q})J$, $(b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}G_{l,\beta,q} - z^{(l,\beta,q,\xi)})N$ и $a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)} - y_{l,\beta,q,\xi}$.

Первая из этих величин равна нулю для любого вектора J, если вектор $\eta^{(l,\beta,q,\xi)}$ определить как решение системы уравнений $\eta^{(l,\beta,q,\xi)}F_{l,\beta,q} = \tau^{(\beta,\xi)}$. Однако система может не иметь решения, поскольку число неизвестных (координат вектора $\eta^{(l,\beta,q,\xi)}$) меньше либо равно числу уравнений ($\nu_l \le n_\beta$). Для этого представим вектор J в виде $J = J^{\perp} + J^{(0)}$, где $J^{\perp} = \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} u_i^{\perp}$, $J^{(0)} = \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i u_i$.

Тогда

$$(\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q})J = (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q})(J^{\perp} + J^{(0)}) =$$

$$= (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q})J^{\perp} + \tau^{(\beta,\xi)}J^{(0)} =$$

$$= (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q}) + \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} u_i^{\perp} + \tau^{(\beta,\xi)} \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i u_i =$$

$$= \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} (\tau^{(\beta,\xi)} u_i^{\perp} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q}u_i^{\perp}) + \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i.$$

Число неизвестных системы $\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}u_i^{\perp} = \tau^{(\beta,\xi)}u_i^{\perp}$ больше либо равно числу уравнений ($\nu_l \ge \rho_{l,\beta,q}$). Система всегда имеет решение: ранг расширенной матрицы равен рангу матрицы системы.

Таким образом,

$$t_{\xi}^{(\beta)}(J) - d_{\xi}^{(l,\beta,q)}(\overline{F}_{l,\beta,q}(J)) = \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} (\tau^{(\beta,\xi)} u_i^{\perp} - \eta^{(l,\beta,q,\xi)}F_{l,\beta,q}u_i^{\perp}) +$$

$$+ \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i + (b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)}G_{l,\beta,q} - z^{(l,\beta,q,\xi)})N +$$

$$+ a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)} - y_{l,\beta,q,\xi}.$$

Равенство (7) выполняется для любого $J \in V_{l,\beta,q}(F)$, если компоненты каждого вектора $\eta^{(l,\beta,q,\xi)}$ определить как решение системы $\rho_{l,\beta,q}$ уравнений с ν_l неизвестными:

$$\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}u_i^{\perp} = \tau^{(\beta,\xi)}u_i^{\perp},\ 1 \le i \le \rho_{l,\beta,q}, \qquad (11)$$

матрицу $z^{(l,\beta,q,\xi)}$ и функции $y_{l,\beta,q,\xi}(F)$ задать равенствами (9) и (10) соответственно. Заметим, что систему уравнений (11) можно записать в виде (8) и для каждого фиксированного F векторы $(\lambda_1,\dots,\lambda_{n_\beta-\rho_{l,\beta,q}})$ принадлежат множеству $\Lambda_{l,\beta,q}(F)$. ■

В частном случае (без учета вектора внешних переменных) теорема 1 доказана в работе [26].

Пример 4 (продолжение примера 3). Найдем функции использования массивов на итерациях преобразованного гнезда циклов. Отметим, что преобразование осуществлялось согласно двумерному таймированию $\bar{t}^{(1)}(1)=(1,1)$; $\bar{t}^{(2)}(i)=(i,1)$, $2\le i\le N$; $\bar{t}^{(3)}(i,j)=(i,j)$, $2\le i\le N$, $1\le j\le i-1$.

Для вхождения $(x,S_1,1)$ и $(x,S_2,1)$ согласно равенству (7) получим $\bar{d}^{(x,S_1,1)}(1)=(1,1)$ и $\bar{d}^{(x,S_2,1)}(i)=(i,1)$.

Рассмотрим вхождения $(x,S_3,1)$ и $(x,S_3,2)$. Достаточно взять только одно из этих вхождений. Имеем $\tau^{(3,1)}=(1,0)$, $\tau^{(3,2)}=(0,1)$, $b^{(3,1)}=b^{(3,2)}=0$, $a_{3,1}=a_{3,2}=0$; $F_{x,S_3,1}=(1\ 0)$, $G_{x,S_3,1}=0$, $f^{(x,S_3,1)}=0$; $\rho_{x,S_3,1}=1$, $n_3=2$; $u_1=(0,1)$, $u_1^{\perp}=(1,0)$. Для $\xi=1$ система (8) примет следующий вид: $\eta^{(x,S_3,1,1)}(1\ 0)\begin{pmatrix}1\\0\end{pmatrix}=$ $=(1\ 0)\begin{pmatrix}1\\0\end{pmatrix}$, отсюда $\eta^{(x,S_3,1,1)}=1$. Из соотношений (9) и (10) получим $z^{(x,S_3,1,1)}=0-1\cdot 0=0$ и $y_{x,S_3,1,1}(i)=0-1\cdot 0+\lambda_1(1,0)(0,1)=0$. Для $\xi=2$ получим $\eta^{(x,S_3,1,2)}(1\ 0)\begin{pmatrix}1\\0\end{pmatrix}=(1\ 0)\begin{pmatrix}1\\0\end{pmatrix}$, отсюда $\eta^{(x,S_3,1,2)}=0$; $z^{(x,S_3,1,2)}=0-0\cdot 0=0$ и $y_{x,S_3,1,2}(i)=0-0\cdot 0+\lambda_1(0,1)(0,1)=\lambda_1$, $\lambda_1\in\Lambda_{x,S_3,1}(i)$. Для каждого $i=1,\dots,N$ определим множество $V_{x,S_3,1}(i)=\{(i,j)\,|\,1\le j\le i-1\}$. Следовательно, $\Lambda_{x,S_3,1}(i)=\{j\,|\,1\le j\le i-1\}$ и $y_{x,S_3,1,2}(i)=j$, $1\le j\le i-1$. Получим $\bar{d}^{(x,S_3,1)}(i)=$ $=\bar{d}^{(x,S_3,2)}(i)=(i,j)$, $1\le j\le i-1$.

Рассмотрим вхождение $(x,S_3,3)$. Имеем $F_{x,S_3,3}=(0\ 1)$, $G_{x,S_3,3}=0$, $f^{(x,S_3,3)}=0$; $\rho_{x,S_3,3}=1$; $u_i=(1,0)$, $u_1^{\perp}=(0,1)$. Для $\xi=1$ получим $\eta^{(x,S_3,3,1)}(0\ 1)\begin{pmatrix}0\\1\end{pmatrix}=(1\ 0)\begin{pmatrix}0\\1\end{pmatrix}$, откуда $\eta^{(x,S_3,3,1)}=0$; $z^{(x,S_3,3,1)}=0-0\cdot 0=0$ и $y_{x,S_3,3,1}(j)=0-0+\lambda_1(1,0)(1,0)=\lambda_1$, $\lambda_1\in\Lambda_{x,S_3,3}(j)$. Для каждого $i=1,\dots,N-1$ определим множество $V_{x,S_3,3}(j)=\{(i,j)\,|\,j+1\le i\le N\}$. Следовательно, $\Lambda_{x,S_3,3}(j)=\{i\,|\,j+1\le i\le N\}$ и $y_{x,S_3,1,1}(j)=i$, $j+1\le i\le N$. Для $\xi=2$ получим $\eta^{(x,S_3,3,2)}(0\ 1)\begin{pmatrix}0\\1\end{pmatrix}=(0\ 1)\begin{pmatrix}0\\1\end{pmatrix}$, откуда $\eta^{(x,S_3,3,2)}=1$; $z^{(x,S_3,3,2)}=0-1\cdot 0=0$ и $y_{x,S_3,3,2}(j)=0-0+\lambda_1(0,1)(1,0)=0$. Таким образом, $\bar{d}^{(x,S_3,2)}(j)=(i,j)$, $j+1\le i\le N$.

Получим функции использования массива b. Для вхождения $(b,S_1,1)$ и $(b,S_2,1)$ согласно равенству (7) найдем $\bar{d}^{(b,S_1,1)}(1)=(1,1)$ и $\bar{d}^{(b,S_2,1)}(i)=(i,1)$.

Определим функцию использования массива a (вхождение $(a,S_3,1)$). Имеем: $\tau^{(3,1)}=(1,0)$, $\tau^{(3,2)}=(0,1)$, $b^{(3,1)}=b^{(3,2)}=0$, $a_{3,1}=a_{3,2}=0$; $F_{a,S_3,1}=E^{(2)}$,

$G_{a,S_3,1} = (0\ 0)^{\mathrm{T}}$, $f^{(a,S_3,1)} = (0,0)$; $\rho_{a,S_3,1} = \operatorname{rank} E^{(2)} = 2$, $n_3 = 2$; u_i отсутствуют, $u_1^{\perp} = (1,0)$, $u_2^{\perp} = (0,1)$. Для $\xi = 1$ система (8) примет следующий вид: $\eta^{(a,S_3,1,1)} E^{(2)} E^{(2)} = (1\ 0) E^{(2)}$, отсюда $\eta^{(a,S_3,1,1)} = (1,0)$. Из соотношений (9) и (10) получим $z^{(a,S_3,1,1)} = 0 - (1\ 0)(0\ 0)^{\mathrm{T}} = 0$ и $y_{a,S_3,1,1}(i,j) = 0 - (1\ 0)(0\ 0)^{\mathrm{T}} = 0$. Аналогично для $\xi = 2$ имеем $\eta^{(a,S_3,1,2)} E^{(2)} E^{(2)} = (0\ 1) E^{(2)}$, откуда $\eta^{(a,S_3,1,2)} = (0,1)$, $z^{(a,S_3,1,2)} = 0 - (0\ 1)(0\ 0)^{\mathrm{T}} = 0$ и $y_{a,S_3,1,2}(i,j) = 0 - (0\ 1)(0\ 0)^{\mathrm{T}} = 0$. Соответственно получаем $\bar{d}^{(a,S_3,1)}(i,j) = (i,j)$. ■

Теорема 2. Если $\rho_{l,\beta,q} < n_\beta$, то каждый элемент массива, связанный с вхождением (l,β,q), используется на итерациях $\bar{d}^{(l,\beta,q)}(F)$, отличающихся линейными комбинациями векторов $T^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}$. Если $\rho_{l,\beta,q} = n_\beta$, то элемент массива используется на фиксированной итерации.

Доказательство. Покажем сначала, что для любого значения $\eta^{(l,\beta,q,\xi)}$, являющегося решением системы (8), при фиксированном F величина $\eta^{(l,\beta,q,\xi)} F - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q} N - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)}$ принимает одно и то же значение. Любой вектор J, для которого $F_{l,\beta,q} J + G_{l,\beta,q} N + f^{(l,\beta,q)} = F$, можно представить в виде $J = J^{\perp} + J^{(0)}$, где $J^{(0)} \in \ker F_{l,\beta,q}$, $J^{\perp}$ — нормальное решение системы уравнений $F_{l,\beta,q} J = F - G_{l,\beta,q} N - f^{(l,\beta,q)}$, $J^{\perp} = \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} u_i^{\perp}$. С учетом доказательства теоремы 1 получим значение

$$\eta^{(l,\beta,q,\xi)} F - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q} N - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} =$$

$$= \eta^{(l,\beta,q,\xi)} (F_{l,\beta,q} J + G_{l,\beta,q} N + f^{(l,\beta,q)}) - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q} N - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} =$$

$$= \eta^{(l,\beta,q,\xi)} F_{l,\beta,q} J = \eta^{(l,\beta,q,\xi)} F_{l,\beta,q} (J^{\perp} + J^{(0)}) =$$

$$= \eta^{(l,\beta,q,\xi)} F_{l,\beta,q} \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} u_i^{\perp} = \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} \eta^{(l,\beta,q,\xi)} F_{l,\beta,q} u_i^{\perp} =$$

$$= \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} \tau^{(\beta,\xi)} u_i^{\perp} = \tau^{(\beta,\xi)} J^{\perp},$$

которое является постоянной величиной.

Согласно теореме 1 координаты функций использования массивов на итерациях гнезда циклов задаются в виде

$$d_{\xi}^{(l,\beta,q)}(F) = \eta^{(l,\beta,q,\xi)} F + (b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q}) N + a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} +$$
$$+ \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i = J^{\perp} + b^{(\beta,\xi)} N + a_{\beta,\xi} + \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i.$$

Следовательно, если $\rho_{l,\beta,q} < n_\beta$, итерации, на которых используется элемент $a_l(F)$, связанный с вхождением (l,β,q), могут отличаться только линейными комбинациями векторов $T^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}$.

Если $\rho_{l,\beta,q} = n_\beta$, сумма $\sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)}$ отсутствует (множество $\Lambda_{l,\beta,q}(F)$ пустое) и значение $d_{\xi}^{(l,\beta,q)}(F)$ становится фиксированным. ■

Введем в рассмотрение множество $\Xi_{fix} = \{\xi_1, \ldots, \xi_\theta\} \subset \{1, \ldots, n\}$, составленное из θ произвольных элементов множества $\{1, \ldots, n\}$. Выделим θ координат многомерного таймирования $t^{(\beta)}_{\xi_i}$, $\xi_i \in \Xi_{fix}$, и обозначим $\bar{t}^{(\beta)}_{fix}$ векторную функцию, составленную из выделенных координат $t^{(\beta)}_{\xi_1}, \ldots, t^{(\beta)}_{\xi_\theta}$.

Обозначим $T^{(\beta)}_{fix}$ матрицу, строки которой составлены из векторов $\tau^{(\beta,\xi_i)}$, $\xi_i \in \Xi_{fix}$; $\rho^{fix}_{l,\beta,q} = \operatorname{rank}\begin{pmatrix} F_{l,\beta,q} \\ T^{(\beta)}_{fix} \end{pmatrix}$. Обозначим u^{fix}_i, $1 \le i \le n_\beta - \rho^{fix}_{l,\beta,q}$, базисные векторы пересечения подпространства $\ker\begin{pmatrix} F_{l,\beta,q} \\ T^{(\beta)}_{fix} \end{pmatrix}$ и $\mathbf{Z}^{n_\beta}$; векторы u^{fix}_i могут отсутствовать. Рассмотрим базис пространства $\mathbf{Z}^{n_\beta}$ с базисными векторами $u^{\perp,fix}_i$, $1 \le i \le \rho^{fix}_{l,\beta,q}$, u^{fix}_i, $1 \le i \le n_\beta - \rho^{fix}_{l,\beta,q}$. Матрицу, столбцы которой составлены из векторов $u^{\perp,fix}_i$, обозначим $U^{\perp,fix}_{l,\beta,q}$.

Пусть T_{fix} — произвольный фиксированный элемент множества значений функции $\bar{t}^{(\beta)}_{fix}$; введем в рассмотрение множество

$$V^{fix}_\beta(T_{fix}) = \{J \in V_\beta \mid \bar{t}^{(\beta)}_{fix}(J) = T_{fix}\}.$$

Рассмотрим множество $V_{l,\beta,q}(F) \cap V^{fix}_\beta(T_{fix})$ итераций исходного гнезда циклов, на вхождении (l, β, q) которых используется одно и то же данное $a_l(F)$ и которые реализуются в преобразованном гнезде циклов на итерациях с фиксированным значением функции $\bar{t}^{(\beta)}_{fix}$, равным T_{fix}. Обозначим $\Lambda^{fix}_{l,\beta,q}(F, T_{fix})$ множество, задающее координаты проекции множества $V_{l,\beta,q}(F) \cap V^{fix}_\beta(T_{fix})$ на линейную оболочку векторов u^{fix}_i:

$$\Lambda^{fix}_{l,\beta,q}(F, T_{fix}) = \left\{ (\lambda^{fix}_1, \ldots, \lambda^{fix}_{n_\beta - \rho^{fix}_{l,\beta,q}}) \mid J = \right.$$
$$\left. = J^{\perp,fix} + \sum_{i=1}^{n_\beta - \rho^{fix}_{l,\beta,q}} \lambda^{fix}_i u^{fix}_i,\ J \in V_{l,\beta,q}(F) \cap V^{fix}_\beta(T_{fix}) \right\};$$

в случае $\rho^{fix}_{l,\beta,q} = n_\beta$ множество $\Lambda^{fix}_{l,\beta,q}(F, T_{fix})$ пустое.

Теорема 3. Итерации преобразованного гнезда циклов, на которых используется элемент массива $a_l(F)$, $F \in W_{l,\beta,q}$, при фиксированном значении функции $\bar{t}^{(\beta)}_{fix}$, равном T_{fix}, определяются значениями функций $d^{(l,\beta,q)}_\xi$, которые задаются формулами (6), где $\eta^{(l,\beta,q,\xi)}$ — решение системы уравнений

$$\eta^{(l,\beta,q,\xi)} F_{l,\beta,q} U^{\perp,fix}_{l,\beta,q} = \tau^{(\beta,\xi)} U^{\perp,fix}_{l,\beta,q}, \tag{12}$$

а матрицы $z^{(l,\beta,q,\xi)}$ и величины $y_{l,\beta,q,\xi}(F)$ определяются соответственно равенствами (9) и следующим равенством:

$$y_{l,\beta,q,\xi}(F) = a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} + \sum_{i=1}^{n_\beta - \rho^{fix}_{l,\beta,q}} \lambda^{fix}_i \tau^{(\beta,\xi)} u^{fix}_i, \tag{13}$$
$$(\lambda^{fix}_1, \ldots, \lambda^{fix}_{n_\beta - \rho^{fix}_{l,\beta,q}}) \in \Lambda^{fix}_{l,\beta,q}(F, T_{fix}).$$

Теорема 4. Если $\rho^{fix}_{l,\beta,q} < n_\beta$, то в преобразованном гнезде циклов при фиксированном значении функции $\bar{t}^{(\beta)}_{fix}$ каждый элемент массива, связанный с вхождением (l, β, q), используется на итерациях, отличающихся линейными комбинациями векторов $T^{(\beta)}_{fix} u^{fix}_i, 1 \le i \le n_\beta - \rho^{fix}_{l,\beta,q}$, где $T^{(\beta)}_{fix}$ — матрица, строки которой составлены из векторов $\tau^{(\beta,\xi_i)}$, $\xi_i \in \{1, \dots, n\} \setminus \Xi_{fix}$. Если $\rho^{fix}_{l,\beta,q} = n_\beta$, то при фиксированном значении функции $\bar{t}^{(\beta)}_{fix}$ элемент массива используется однократно.

Доказательство теорем 3 и 4 по сути повторяет доказательство теорем 1 и 2 соответственно, следует лишь зафиксировать F и T_{fix}.

РАСПРЕДЕЛЕНИЕ ОПЕРАЦИЙ И ДАННЫХ МЕЖДУ ПРОЦЕССОРАМИ И ИТЕРАЦИЯМИ

Введем в рассмотрение множество $\Xi_s = \{\xi_1, \dots, \xi_r\} \subset \{1, \dots, n\}$, составленное из r произвольных элементов множества $\{1, \dots, n\}$, и множество $\Xi_t = \{\xi_{r+1}, \dots, \xi_n\} \subset \{1, \dots, n\} \setminus \Xi_s$. Для удобства использования будем считать множества Ξ_s и Ξ_t упорядоченными: $\xi_i < \xi_j$, если $i < j$ (независимо для любых $\xi_i, \xi_j \in \Xi_s$ и $\xi_i, \xi_j \in \Xi_t$).

Введем в рассмотрение следующие векторные функции: $\bar{t}^{(\beta)}_s$, составленную из функций $t^{(\beta)}_{\xi_i}$, $\xi_i \in \Xi_s$, и $\bar{t}^{(\beta)}_t$, составленную из функций $t^{(\beta)}_{\xi_i}$, $\xi_i \in \Xi_t$; $\bar{d}^{(l,\beta,q)}_s$, составленную из функций $d^{(l,\beta,q)}_{\xi_i}$, $\xi_i \in \Xi_s$, и $\bar{d}^{(l,\beta,q)}_t$, составленную из функций $d^{(l,\beta,q)}_{\xi_i}$, $\xi_i \in \Xi_t$.

Для реализации алгоритма на параллельном компьютере будем полагать, что функции $\bar{t}^{(\beta)}_s$ задают отображение операций алгоритма в r-мерное пространство виртуальных процессоров, а функции $\bar{t}^{(\beta)}_t$ задают итерации, выполняемые процессорами. Функции $\bar{d}^{(l,\beta,q)}_s$ задают координаты процессоров, в которых используются элементы массивов, связанные с вхождением (l, β, q), а функции $\bar{d}^{(l,\beta,q)}_t$ задают итерации, на которых эти элементы используются.

С помощью теоремы 1 можно найти распределение входных и выходных данных между процессорами, получить информацию об объеме массивов, используемых в каждом процессоре.

Распределение между виртуальными процессорами входных массивов можно задать с помощью функций $\bar{d}^{(l,\beta,q)}_s$. Для этого элемент $a_l(F)$ входного массива a_l, $F \in W_{l,\beta,q}$, следует распределить в процессор $P_{in}(\bar{d}^{(l,\beta,q)}_s(F))$, координатами которого являются значения функций $d^{(l,\beta,q)}_{\xi_i}(F)$, $\xi_i \in \Xi_s$, соответствующие вектору $(d^{(l,\beta,q)}_1(F), \dots, d^{(l,\beta,q)}_{\xi_r}(F))$ с лексикографически наименьшим значением.

Результаты вычислений — элементы $a_l(F)$ выходного массива — находятся в процессоре $P_{out}(\bar{d}^{(l,\beta,q)}_s(F))$, координатами которого являются значения функций $d^{(l,\beta,q)}_{\xi_i}(F)$, $\xi_i \in \Xi_s$, соответствующие вектору $(d^{(l,\beta,q)}_1(F), \dots, d^{(l,\beta,q)}_{\xi_r}(F))$ с лексикографически наибольшим значением.

Функции $\bar{d}^{(l,\beta,q)}_s$ позволяют также получить информацию об объеме массивов, используемых в каждом процессоре. Чтобы определить элементы $a_l(F)$ вы-

ходного массива a_l, используемые в процессоре с координатами $(t_1, \dots, t_r)$, следует найти все элементы F множества $W_{l,\beta,q}$, для которых $(d_{\xi_1}^{(l,\beta,q)}(F), \dots, d_{\xi_r}^{(l,\beta,q)}(F)) = (t_1, \dots, t_r)$.

Введем обозначения: $T_s^{(\beta)}$ — матрица, строки которой составлены из векторов $\tau^{(\beta,\xi_i)}$, $\xi_i \in \Xi_s$; $T_t^{(\beta)}$ — матрица, строки которой составлены из векторов $\tau^{(\beta,\xi_i)}$, $\xi_i \in \Xi_t$; $\rho_{l,\beta,q}^s = \operatorname{rank}\begin{pmatrix} F_{l,\beta,q} \\ T_s^{(\beta)} \end{pmatrix}$, $\rho_{l,\beta,q}^t = \operatorname{rank}\begin{pmatrix} F_{l,\beta,q} \\ T_t^{(\beta)} \end{pmatrix}$.

Сформулируем следствия из теоремы 2, позволяющие выявить, требуется ли передача данных между процессорами и итерациями параллельного алгоритма либо данные используются одним процессором или на одной итерации, а также задать векторы, определяющие направление передачи.

Следствие 1. Если $\rho_{l,\beta,q} < \rho_{l,\beta,q}^s$, то каждый элемент массива, связанный с вхождением (l, β, q), используется виртуальными процессорами, коммуникации между которыми можно задать векторами $T_s^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}$; если $\rho_{l,\beta,q} = \rho_{l,\beta,q}^s$, то элемент массива используется только одним процессором.

Действительно, из доказательства теоремы 2 следует, что для каждого элемента массива, связанного с вхождением (l, β, q), значения $\bar{d}_s^{(l,\beta,q)}(F)$ отличаются линейными комбинациями векторов $T_s^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}^s$. Если $\rho_{l,\beta,q} = \rho_{l,\beta,q}^s$, то $\ker\begin{pmatrix} F_{l,\beta,q} \\ T_s^{(\beta)} \end{pmatrix} = \ker F_{l,\beta,q}$, $T_s^{(\beta)} u_i = 0$ для всех u_i. Следовательно, в этом случае $\bar{d}_s^{(l,\beta,q)}(F)$ принимает фиксированное значение, т.е. элемент массива используется только одним процессором.

Следствие 2. Если $\rho_{l,\beta,q} < \rho_{l,\beta,q}^t$, то каждый элемент массива, связанный с вхождением (l, β, q), используется на итерациях, отличающихся линейными комбинациями векторов $T_t^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}$; если $\rho_{l,\beta,q} = \rho_{l,\beta,q}^t$, то элемент массива используется только на одной итерации.

Доказательство следствия 2 аналогично доказательству следствия 1.

Обозначим u_i^s, $1 \le i \le n_\beta - \rho_{l,\beta,q}^s$, базисные векторы пересечения подпространства $\ker\begin{pmatrix} F_{l,\beta,q} \\ T_s^{(\beta)} \end{pmatrix}$ и $\mathbf{Z}^{n_\beta}$, через u_i^t, $1 \le i \le n_\beta - \rho_{l,\beta,q}^t$, — базисные векторы пересечения подпространства $\ker\begin{pmatrix} F_{l,\beta,q} \\ T_t^{(\beta)} \end{pmatrix}$ и $\mathbf{Z}^{n_\beta}$; векторы u_i^s, u_i^t могут отсутствовать.

Сформулируем следствия из теоремы 4, позволяющие выявить характер использования данных фиксированным процессором или на фиксированной итерации параллельного алгоритма.

Следствие 3. Если $\rho_{l,\beta,q}^s < n_\beta$, то в одном виртуальном процессоре каждый элемент массива, связанный с вхождением (l, β, q), используется на итерациях, отличающихся линейными комбинациями векторов $T_t^{(\beta)} u_i^s$, $1 \le i \le n_\beta - \rho_{l,\beta,q}^s$; если $\rho_{l,\beta,q}^s = n_\beta$, то элемент массива используется в одном процессоре только на одной итерации.

Следствие 4. Если $\rho^t_{l,\beta,q} < n_\beta$, то на одной итерации каждый элемент массива, связанный с вхождением (l, β, q), используется виртуальными процессорами, коммуникации между которыми можно задать векторами $T_s^{(\beta)} u_i^t$, $1 \le i \le n_\beta - \rho^t_{l,\beta,q}$; если $\rho^t_{l,\beta,q} = n_\beta$, то элемент массива используется на одной итерации только в одном процессоре.

Следствия 3 и 4 можно получить, если в условиях теоремы 4 положить $\Xi_{fix} = \Xi_s$ и $\Xi_{fix} = \Xi_t$ соответственно.

ОРГАНИЗАЦИЯ ОБМЕНА ДАННЫМИ

На практике даже оптимальное начальное распределение данных не исключает необходимости обменом между процессорами в процессе выполнения алгоритма. Как вытекает из следствия 1, обмен данными может быть необходимым для элементов массива a_l, связанных с вхождением (l, β, q), если имеет место $\rho^s_{l,\beta,q} > \rho_{l,\beta,q}$.

Известно, что на параллельных компьютерах с распределенной памятью такие структурированные коммуникации, как бродкаст (broadcast), разброс (scatter), сборка (gather), редукция (reduction), а также трансляция (translation) данных выполняются значительно быстрее, чем соответствующий набор коммуникаций точка–точка (point–to–point). Поэтому желательно выявлять возможность организации таких коммуникаций.

Сформулируем и исследуем условия, позволяющие в некоторых случаях определять возможность организации наиболее часто используемых быстрых коммуникаций — бродкаста и трансляции данных. Бродкаст (одновременное распространение) — это передача данного группе процессоров, в которых данное на одной итерации используется как аргумент. Трансляция — это передача данного от одного процессора к другому в случае, если элемент массива используется в разных процессорах по очереди. Коммуникация точка–точка — это передача данного от одного процессора к другому.

Организация бродкаста. Пусть имеются функции вида (6), задающие использование массивов данных виртуальными процессорами, и n-мерное таймирование с координатами вида (3), из которых r координат задают пространственное отображение операций алгоритма в r-мерное пространство виртуальных процессоров, а остальные $n - r$ координат задают итерации, выполняемые процессорами в лексикографическом порядке.

Пусть вектор T является элементом множества значений функции $\bar{t}_t^{(\beta)}$. Рассмотрим множество $V^t_\beta(T) = \{ J \in V_\beta \mid \bar{t}_t^{(\beta)}(J) = T \}$ итераций исходного гнезда циклов, на которых выполняется оператор S_β и которые реализуются в пространстве виртуальных процессоров на итерации T, и множество $V_{l,\beta,q}(F) \cap V^t_\beta(T)$ итераций исходного гнезда циклов, на вхождении (l, β, q) которых используется одно и то же данное $a_l(F)$ и которые реализуются в пространстве виртуальных процессоров на итерации T.

Обозначим $U^t_{l,\beta,q}$ матрицу, столбцы которой составлены из векторов u_i^t.

Лемма 1 [18]. Пусть существуют истинные зависимости, порождаемые вхождением (l, β, q); $\bar{\Phi}_{\alpha,\beta}$ — функция зависимостей. Если выполняется условие

$$\Phi_{\alpha,\beta} U^t_{l,\beta,q} = 0, \tag{14}$$

то между итерациями множества $V_{l,\beta,q}(F) \cap V^t_\beta(T)$ данное $a_l(F)$ не переопределяется.

Теорема 5. Пусть $\rho^t_{l,\beta,q} < n_\beta$. На итерации T можно организовать бродкаст данного $a_l(F)$ к процессорам $P(\bar{d}^{(l,\beta,q)}_s(F))$, где параметры функций $\bar{d}^{(l,\beta,q)}_s$ задаются согласно (12), (9) и (13), полагая $\Xi_{fix} = \Xi_t$, $T_{fix} = T$, в одном из следующих случаев:

1) элементы массива a_l встречаются только в правых частях операторов алгоритма;

2) для истинной зависимости, порожденной вхождением (l, β, q), выполняется условие (14).

Доказательство. Из условия теоремы следует, что функция $\bar{F}_{l,\beta,q}$ встречается в правой части оператора S_β, поэтому элементы массива a_l используются на q-м вхождении в оператор S_β в качестве аргументов. Так как $\rho^t_{l,\beta,q} < n_\beta$, то согласно следствию 4 на одной итерации каждый элемент массива, связанный с вхождением (l, β, q), используется виртуальными процессорами, коммуникации между которыми можно задать векторами $T^{(\beta)}_s u^t_i$, $1 \le i \le n_\beta - \rho^t_{l,\beta,q}$; координаты таких процессоров согласно теореме 3 при $T_{fix} = T$, $\Xi_{fix} = \Xi_t$ задаются функцией $\bar{d}^{(l,\beta,q)}_s(F)$, параметры которой определяются из системы (12) и соотношений (9) и (13). Процессоры используют одно и то же значение данного $a_l(F)$, поскольку между итерациями множества $V_{l,\beta,q}(F) \cap V^t_\beta(T)$ это данное не переопределяется: в случае 1 это данное не переопределяется вообще, в случае 2 невозможность переопределения гарантирована условием (14). ■

В случае 1 теоремы 5 бродкаст элемента $a_l(F)$ осуществляется от процессора $P_{in}(\bar{d}^{(l,\beta,q)}_s(F))$, в который распределяется $a_l(F)$, если не применялась другая схема распределения входных массивов (например, репликация массивов). В случае 2 бродкаст осуществляется от процессора, в котором $a_l(F)$ вычислялся.

Заметим, что при выполнении условий теоремы 5 бродкаст может быть вырожденным, если в силу особенностей множества V_β множество $V_{l,\beta,q}(F) \cap V^t_\beta(T)$ содержит только один элемент.

Организация трансляции данных. Сформулируем и докажем теоремы, которые позволяют выявлять возможность организации трансляции данного. В теореме 6 будет исследован случай, когда данное используется как аргумент на разных итерациях разными процессорами и передача данного осуществляется на нескольких итерациях. В теореме 7 будет исследован случай, когда данное используется как аргумент и переопределяется разными процессорами по очереди на одной итерации со сдвигом по времени при выполнении программы.

Пусть $\rho_{l,\beta,q} < n_\beta$. Обозначим $U_{l,\beta,q}$ матрицу, столбцы которой составлены из базисных векторов u_i, $1 \le i \le n_\beta - \rho_{l,\beta,q}$.

Лемма 2 [18]. Пусть существуют истинные зависимости, порождаемые вхождением (l, β, q); $\bar{\Phi}_{\alpha,\beta}$ — функция зависимостей. Если выполняется условие

$$\Phi_{\alpha,\beta} U_{l,\beta,q} = 0, \tag{15}$$

то между итерациями множества $V_{l,\beta,q}(F)$ данное $a_l(F)$ не переопределяется.

Теорема 6. Пусть $\rho^t_{l,\beta,q} > \rho_{l,\beta,q}$, $\rho^s_{l,\beta,q} > \rho_{l,\beta,q}$, $\rho^t_{l,\beta,q} = n_\beta$. На итерациях $\bar{d}^{(l,\beta,q)}_t(F)$ можно организовать трансляцию данного $a_l(F)$ между процессора-

ми $P(\bar{d}_s^{(l,\beta,q)}(F))$, где параметры функций $\bar{d}_s^{(l,\beta,q)}$ и $\bar{d}_t^{(l,\beta,q)}$ задаются согласно (8), (9) и (10) в одном из следующих случаев:

1) массив a_l встречается только в правых частях операторов алгоритма;

2) для истинной зависимости, порожденной вхождением (l, β, q), выполняется условие (15).

Доказательство. Функция $\bar{F}_{l,\beta,q}$ встречается в правой части оператора S_β, поэтому элементы массива a_l используются на q-м вхождении в оператор S_β в качестве аргументов. Условие $\rho^t_{l,\beta,q} > \rho_{l,\beta,q}$ означает, что данное $a_l(F)$ во время выполнения программы используется на q-м вхождении массива a_l в оператор S_β на нескольких итерациях (следствие 2), а условие $\rho^s_{l,\beta,q} > \rho_{l,\beta,q}$ означает, что $a_l(F)$ используется в нескольких процессорах (следствие 1); из условия $\rho^t_{l,\beta,q} = n_\beta$ вытекает (следствие 4), что на фиксированной итерации массив a_l используется только одним виртуальным процессором.

Таким образом, данное $a_l(F)$ используется на нескольких итерациях $\bar{d}_t^{(l,\beta,q)}(F)$, в нескольких процессорах $P(\bar{d}_s^{(l,\beta,q)}(F))$ и на каждой итерации данное используется одним процессором. Параметры функций согласно теореме 1 определяются из системы (8) и соотношений (9) и (10). Процессоры используют одно и то же данное, поскольку между итерациями множества $V_{l,\beta,q}(F)$ данное $a_l(F)$ не переопределяется: в случае 1 это данное не переопределяется вообще, в случае 2 невозможность переопределения гарантирована условием (15). ■

Трансляцию следует осуществлять согласно лексикографической упорядоченности векторов $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$. В случае 1 теоремы 6 до начала трансляции данное $a_l(F)$ согласно начальному распределению находится в локальной памяти процессора $P_{in}(\bar{d}_s^{(l,\beta,q)}(F))$, который первым участвует в трансляции. В случае 2 теоремы 6 до начала трансляции следует осуществить передачу данного $a_l(F)$ от процессора, в котором $a_l(F)$ вычислялся, к процессору $P(\bar{d}_s^{(l,\beta,q)}(F))$, координатами которого являются значения функций $d_{\xi_i}^{(l,\beta,q)}(F)$, $\xi_i \in \Xi_s$, соответствующие вектору $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$ с лексикографически наименьшим значением.

Теорема 7. Пусть $\rho^t_{l,\beta,q} < n_\beta$, существует истинная зависимость, порожденная вхождением (l, β, p) в левую часть оператора S_β и вхождением (l, β, q) в правую часть оператора. Тогда на итерации T можно организовать трансляцию данного $a_l(F)$ между процессорами $P(\bar{d}_s^{(l,\beta,q)}(F))$, где параметры функций $\bar{d}_s^{(l,\beta,q)}$ задаются согласно (12), (9) и (13), полагая $\Xi_{fix} = \Xi_t$, $T_{fix} = T$.

Доказательство. Условие $\rho^t_{l,\beta,q} < n_\beta$ означает, что на итерации T оператор S_β выполняется, вообще говоря, более чем в одном процессоре $P(\bar{d}_s^{(l,\beta,q)}(F))$ (следствие 4). Так как существует истинная зависимость, порожденная вхождением p массива a_l в левую часть оператора S_β и вхождением q в правую часть оператора, то данное $a_l(F)$ на итерации T используется как аргумент и переопределяется. Сказанное означает, что на итерации T можно организовать трансляцию данного $a_l(F)$ между процессорами $P(\bar{d}_s^{(l,\beta,q)}(F))$. Координаты таких процессоров согласно теореме 3 при $T_{fix} = T$, $\Xi_{fix} = \Xi_t$ задаются функцией $\bar{d}_s^{(l,\beta,q)}(F)$, параметры которой определяются из системы (12) и соотношений (9) и (13). ■

Трансляцию следует осуществлять согласно лексикографической упорядоченности векторов $(d_1^{(l,\beta,q)}(F),\dots,d_{\xi_r}^{(l,\beta,q)}(F))$. До начала трансляции данного $a_l(F)$ следует осуществить его передачу от процессора, в котором он находился, к процессору $P(\overline{d}_s^{(l,\beta,q)}(F))$, координатами которого являются значения функций $d_{\xi_i}^{(l,\beta,q)}(F)$, $\xi_i \in \Xi_s$, соответствующие вектору $(d_1^{(l,\beta,q)}(F),\dots,d_{\xi_r}^{(l,\beta,q)}(F))$ с лексикографически наименьшим значением.

Как и в случае бродкаста, трансляция может быть вырожденной, если множество $V_{l,\beta,q}(F)\cap V_\beta^t(T)$ содержит только один элемент.

Пример 5 (продолжение примера 4). В соответствии с найденными ранее функциями использования массивов установим распределение элементов массивов по процессорам и итерациям параллельного алгоритма и определим схему обмена данными.

Рассмотрим случай $\Xi_s=\{1\}$, $\Xi_t=\{2\}$. Первая координата многомерного таймирования задает отображение операций алгоритма на линейку процессоров, вторая координата задает порядок выполнения операций процессорами. Запишем соответствующий код (единым образом для каждого процессора), предназначенный для выполнения алгоритма на N процессорах. Символ p обозначает номер процессора. Переменная цикла t соответствует итерациям алгоритма

```
if (1 ≤ p ≤ N)
  do t = 1, max (1, p − 1)
    if (p = t = 1)      S1 : x(1) = b(1)
    if (p ≥ 2, t = 1)   S2 : x(p) = b(p)
    if (p ≥ 2)          S3 : x(p) = x(p) − a(p, t)x(t)
  enddo
endif
```

Имеем: $T_s^{(1)}=(0)$, $T_s^{(2)}=(1)$, $T_s^{(3)}=(1,0)$; $T_t^{(1)}=(0)$, $T_t^{(2)}=(0)$, $T_t^{(3)}=(0,1)$. Равенства $\rho_{a_l,S_\beta,q}=\rho^s_{a_l,S_\beta,q}$ выполняются для всех вхождений (a_l,S_β,q), кроме $(x,S_3,3)$. Согласно следствию 1 каждый элемент массива, связанный с любым вхождением (a_l,S_β,q), за исключением $(x,S_3,3)$, используется только одним процессором. Для использования массива x на третьем его вхождении в оператор S_3 необходимо осуществлять передачу данных.

Истинные зависимости, порожденные третьим вхождением массива x в оператор S_3, задаются функциями $\overline{\Phi}_{1,3}$ и $\overline{\Phi}^{(1)}_{3,3}$. Для обеих функций выполняются условия (14): $\Phi_{1,3}U^t_{x,S_3,3}=(0\ 0)\begin{pmatrix}1\\0\end{pmatrix}=0$, $\Phi^{(1)}_{3,3}U^t_{x,S_3,3}=\begin{pmatrix}0&1\\0&1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}=0$. Следовательно, согласно теореме 5 ($\rho^t_{x,S_3,3}=1$, $n_3=2$, $\rho^t_{x,S_3,3}<n_3$) на итерации T можно осуществить бродкаст данного $x(t)$ к процессорам $P(\overline{d}_s^{(x,S_3,3)}(t))=$ $=P(\eta^{(x,S_3,3,1)}t+z^{(x,S_3,3,1)}N+y_{x,S_3,3,1}(t))$, где $\eta^{(x,S_3,3,1)}$, $z^{(x,S_3,3,1)}$ и $y_{x,S_3,3,1}(t)$ находятся по формулам (12), (9) и (13), полагая $\Xi_{fix}=\Xi_t=\{2\}$, $T_{fix}=T$. Получаем $\eta^{(x,S_3,3,1)}(0\ 1)\begin{pmatrix}0\\1\end{pmatrix}=(1\ 0)\begin{pmatrix}0\\1\end{pmatrix}$, откуда $\eta^{(x,S_3,3,1)}=0$; $z^{(x,S_3,3,1)}=0-0\cdot 0=0$ и

$y_{x,S_3,3,1}(t) = 0 - 0 + \lambda_1(1,0)(1,0) = \lambda_1$, $\lambda_1 \in \Lambda^t_{x,S_3,3}(t,T)$. Поскольку $V_{x,S_3,3}(t) = \{(p,t) \mid t+1 \le p \le N\}$ для каждого $t = 1, \ldots, N-1$ и $V^t_3(T) = \{(p,T) \mid T+1 \le p \le N\}$, то $V_{x,S_3,3}(t) \cap V^t_3(T) = \{(p,T) \mid T+1 \le p \le N\}$ для $t = T$, а $V_{x,S_3,3}(t) \cap \cap V^t_3(T) = \varnothing$ для $t \ne T$. Тогда $\Lambda^t_{x,S_3,3}(T,T) = \{p \mid T+1 \le p \le N\}$ и $\Lambda^t_{x,S_3,3}(t,T) = \varnothing$, $t \ne T$. Соответственно $y_{x,S_3,3,1}(T) = p$, где $T+1 \le p \le N$, и $y_{x,S_3,3,1}(t) = 0$, $t \ne T$.

Таким образом, на итерации T бродкаст данного $x(T)$ осуществляется от процессора $P(T)$, в котором $x(T)$ вычислялся, к процессорам $P(p)$, $T+1 \le p \le N$.

Теперь можно записать псевдокод алгоритма (правила размещения коммуникационных операций в коде программы описаны в [18]):

```
if (1 ≤ p ≤ N)
  do t = 1, max(1, p−1)
    if (p = t = 1)     S1 : x(1) = b(1)
    if (p ≥ 2, t = 1)  S2 : x(p) = b(p)
    if (p ≥ 2)
      Broadcast: Receive x(t) from P(t)
        S3 : x(p) = x(p) − a(p, t)x(t)
    endif
  enddo
  if(p ≤ N −1)
    Broadcast: Send x(p) to P(p+1), ..., P(N)
    endif
endif
```

Рассмотрим случай $\Xi_s = \{2\}$, $\Xi_t = \{1\}$ (вторая координата многомерного таймирования задает отображение операций алгоритма на линейку процессоров, первая координата задает порядок выполнения операций процессорами).

Запишем код, предназначенный для выполнения алгоритма на $N-1$ процессоре:

```
do t = 1, N
  if (1 ≤ p ≤ t −1)
    if (t = p = 1)     S1 : x(1) = b(1)
    if (t ≥ 2, p = 1)  S2 : x(p) = b(p)
    if (t ≥ 2)         S3 : x(p) = x(p) − a(p, t)x(t)
  endif
enddo
```

Равенства $\rho_{a_l,S_\beta,q} = \rho^s_{a_l,S_\beta,q}$ выполняются для всех вхождений (a_l, S_β, q), кроме $(x, S_3, 1)$ и $(x, S_3, 2)$ ($\rho_{x,S_3,1} = \rho_{x,S_3,2} = 1$, $\rho^s_{x,S_3,1} = \rho^s_{x,S_3,2} = 2$). При выполнении алгоритма параллельным компьютером для использования элементов массива x на первом и втором вхождениях этого массива в оператор S_3 необходимо осуществлять передачу данных.

Так как $\rho^t_{x,S_3,1} < n_3$ ($\rho^t_{x,S_3,1} = 1$, $n_3 = 2$) и существует истинная зависимость, порожденная вхождениями $(x, S_3, 1)$, $(x, S_3, 2)$, то согласно теореме 7 на итера-

ции T можно организовать трансляцию данного $x(t)$ между процессорами $P(\bar{d}_s^{(x,S_3,1)}(t)) = P(\eta^{(x,S_3,1,2)} t + z^{(x,S_3,1,2)} N + y_{x,S_3,1,2}(t))$, где $\eta^{(x,S_3,1,2)}$, $z^{(x,S_3,1,2)}$ и $y_{x,S_3,1,2}(t)$ находятся по формулам (12), (9) и (13), полагая $\Xi_{fix} = \Xi_t = \{1\}$, $T_{fix} = T$. Получаем $\eta^{(x,S_3,1,2)}(1\ 0)\begin{pmatrix}1\\0\end{pmatrix} = (0\ 1)\begin{pmatrix}1\\0\end{pmatrix}$, откуда $\eta^{(x,S_3,1,2)} = 0$; $z^{(x,S_3,1,2)} = 0 - 0 \cdot 0 = 0$ и $y_{x,S_3,1,2}(t) = 0 - 0 \cdot 0 + \lambda_1 (0,1)(0,1) = \lambda_1$, $\lambda_1 \in \Lambda^t_{x,S_3,1}(t,T)$. Поскольку $V_{x,S_3,1}(t) = \{(t,p) \mid 1 \le p \le t-1\}$ для каждого $t = 1, \ldots, N$ и $V_3^t(T) = \{(T,p) \mid 1 \le p \le T-1\}$, то $V_{x,S_3,1}(t) \cap V_3^t(T) = \{(T,p) \mid 1 \le p \le T-1\}$ для $t = T$ и $V_{x,S_3,1}(t) \cap V_3^t(T) = \varnothing$ для $t \ne T$. Тогда $\Lambda^t_{x,S_3,1}(T,T) = \{p \mid T+1 \le p \le N\}$ и $\Lambda^t_{x,S_3,1}(t,T) = \varnothing$, $t \ne T$. Соответственно $y_{x,S_3,1,2}(T) = p$, где $1 \le p \le T-1$ и $y_{x,S_3,1,2}(t) = 0$, $t \ne T$.

Таким образом, на итерации T трансляция данного $x(T)$ осуществляется между процессорами $P(T), 1 \le p \le T-1$. Запишем псевдокод алгоритма, воспользовавшись правилами установки приема и передачи данных, описанными в [18]:

```
if (2 ≤ p ≤ N −1) Receive x(p) from P(p−1)
do t = 1, N
   if (1 ≤ p ≤ t −1)
      if (t = p = 1)     S1 : x(1) = b(1)
      if (t ≥ 2, p = 1)  S2 : x(t) = b(t)
      if (t ≥ 2)
          if (p ≥ 2) Receive x(t) from P(p−1)
          S3 : x(p) = x(p) − a(p, t)x(t)
          if (p ≤ N −1) Send x(t) to P(p+1)
      endif
   endif
enddo
```

Отметим, что метод организации коммуникаций, предложенный в настоящей статье, является более простым в применении по сравнению с методом работы [18] и не приводит к появлению избыточных коммуникаций.

ЗАКЛЮЧЕНИЕ

В настоящей статье рассматриваются некоторые аспекты отображения алгоритмов, заданных последовательными программами, на параллельные компьютеры с распределенной памятью. Исследуются задачи, возникающие после распределения операций между процессорами и установления порядка выполнения операций параллельным алгоритмом.

Суммируем полученные результаты:

— для каждого элемента массивов данных указаны процессоры и итерации, на которых он используется;

— исследованы задача распределения между процессорами начальных данных (входных массивов) и задача определения объема массивов, используемых в каждом процессоре;

— получены новые удобные в практическом использовании достаточные условия для возможности организации одновременного распространения (broadcast) и трансляции данных (translation).

В дальнейших исследованиях авторы работы предполагают применить разработанный математический аппарат (функции использования элементов массивов) для формализации коммуникационных операций (в исходном алгоритме формализованы только вычислительные операции) и организации динамического распределения элементов массивов при реализации параллельного алгоритма.

СПИСОК ЛИТЕРАТУРЫ

1. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. — С.-Пб.: БХВ-Петербург, 2002. — 608 с.
2. Lim A.W., Lam M.S. An affine partitioning algorithm to maximize parallelism and minimize communication // Proceedings of the 1-st ACM SIGARCH International Conference on Supercomputing, 1999. — P. 228–237.
3. Darte A., Vivien F. Automatic parallelization based on multi-dimensional scheduling // Technical Report 94-24. LIP, ENS-Lion, 1994.
4. Адуцкевич Е.В., Лиходед Н.А. Согласованное получение конвейерного параллелизма и распределения операций и данных между процессорами // Программирование. — 2006. — **32**, № 3. — С. 54–65.
5. Адуцкевич Е.В., Баханович С.В., Лиходед Н.А. Условия получения согласованного таймирования и распределения операций и данных между процессорами // Докл. Междунар. науч. конф. «Суперкомпьютерные системы и их применение» (SSA'2004). — Минск, Республика Беларусь, 26–28 октября 2004. — С. 160–164.
6. Shang W., Fortes J.A.B. Independent partitioning of algorithms with uniform dependencies // IEEE Trans. on Computers. — 1992. — **41**, N 2. — P. 190–206.
7. Solving alignment using elementary linear algebra / D. Bau, I. Kodukula, V. Kotluar, K. Pingali, P. Stodghill // Proceedings of the Seventh Workshop on Languages and Compilers for Parallel Computing. Springer-Verlag, 1994. — P. 46–60.
8. Lim A.W., Lam M.S. Communication-free parallelization via affine transformation // Proceedings of the Seventh Workshop on Languages and Compilers for Parallel Computing. Springer-Verlag, 1994. — P. 92–106.
9. Лиходед Н.А. Отображение аффинных гнезд циклов на независимые процессоры // Кибернетика и системный анализ. — 2003. — № 3. — С. 169–179.
10. Schreiber R., Dongarra J.J. Automatic blocking of nested loops // Technical Report 90-38. The University of Tennessee, 1995.
11. Dion M., Risset T., Robert Y. Resource-constrained scheduling of partitioned algorithms on processors arrays // Integration, the VLSI Journal. — 1996. — **20**, N 2. — P. 139–159.
12. Фролов А.В. Нахождение и использование ориентированных разрезов реальных графов алгоритмов // Программирование. — 1997. — № 4. — С. 71–80.
13. Lim A.W., Liao S.-W., Lam M.S. Blocking and array contraction across arbitrary nested loops using affine partitioning // Proceedings of the ACM SIGPLAN Symposium on Principles and Practice of Programming Languages, 2001. — P. 103–112.
14. Darte A., Robert Y. On the alignment problem // Parallel Processing Letters. — 1994. — **4**, N 3. — P. 259–270.
15. Dion M., Robert Y. Mapping affine loop nests // Parallel Computing. — 1996. — **22**, N 10. — P. 1373–1397.
16. Adutskevich E.V., Likhoded N.A. Mapping affine loop nests: solving of the alignment and scheduling problems // Proc. 7th Int. Conf. on Parallel Computing Technologies (PaCT-2003), Nizhni Novgorod, Russia, Sept. 15–19, 2003. — Berlin: Springer, 2003. — P. 1–9.
17. Адуцкевич Е.В., Баханович С.В. Адаптация алгоритмов к реализации на системах с распределенной памятью. Пространственно-временная локализация и распределение данных // Докл. междунар. науч. конф. «Суперкомпьютерные системы и их применение» (SSA'2004). — Минск, Республика Беларусь, 26–28 октября 2004. — С. 165–169.

18. Адуцкевич Е.В., Лиходед Н.А. Оптимизация обмена данными на параллельных компьютерах с распределенной памятью // Кибернетика и системный анализ. — 2006. — № 2. — С. 166–182.

19. Dion M., Randriamaro C., Robert Y. Compiling affine nested loops: how to optimize the residual communications after the alignment phase? // J. of Parallel and Distrib. Computing. — 1996. — **30**, N 2. — P. 176–187.

20. Garcia J., Ayguade E., Labarta J. A framework for integrating data alignment, distribution, and redistribution in distributed memory multiprocessors // IEEE Transactions on Parallel and Distributed Systems. — 2001. — **12**, N 4. — P. 416–430.

21. Toward automatic data distribution for migrating computations / L. Pan, J. Xue, M.K. Lai, M.B. Dillencourt, L.F. Bic // Int. Conf. on Parallel Processing, Xian, China, 2007.

22. Banerjee U. An introduction to a formal theory of dependence analysis // J. Supercomput. — 1988. — N 2. — P. 133–149.

23. Feautrier P. Some efficient solutions to the affine scheduling problem. Part 1 // International Journal of Parallel Programming. — 1992. — **21**, N 5. — P. 313–348.

24. Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model / U. Bondhugula, M. Baskaran, S. Krishnamoorthy, J. Ramanujam, A. Rountev, P. Sadayappan // Lecture Notes in Computer Science. — 2008. — **4959**. — P. 132–146.

25. Voevodin V.V. Information structure of sequential programs // Russ. J. of Numerical Analysis and Math. Modelling. — 1995. — **10**, N 3. — P. 279–286.

26. Лиходед Н.А. Функции распределения между процессорами и итерациями параллельного алгоритма // Докл. НАН Беларуси. — 2007. — **51**, № 4. — С. 19–24.

Поступила 14.01.2008
После доработки 19.03.2010