ВЫСШАЯ ШКОЛА ЭЛЕКТРОНИКИ И КОМПЬЮТЕРНЫХ НАУК

УДК 004.032.24 + 004.021

Q-ЭФФЕКТИВНАЯ РЕАЛИЗАЦИЯ ЧИСЛЕННЫХ АЛГОРИТМОВ[1]

В.Н. Алеева

Описан подход, основанный на концепции Q-детерминанта, который позволяет для любого численного алгоритма разработать программу, использующую ресурс параллелизма алгоритма полностью. Результаты исследования могут применяться для повышения эффективности реализации численных алгоритмов на параллельных вычислительных системах.

Ключевые слова: Q-детерминант алгоритма, представление алгоритма в форме Q-детерминанта, Q-эффективная реализация алгоритма, ресурс параллелизма алгоритма, параллельная вычислительная система, параллельная программа, Q-эффективная программа.

История развития параллельных вычислительных систем насчитывает десятки лет, однако эффективность реализации на них алгоритмов остается низкой. Использование ресурса параллелизма алгоритмов может решить эту проблему. Выявить ресурс параллелизма численных алгоритмов позволяет концепция Q-детерминанта [1]. Основой концепции является универсальное описание численных алгоритмов и представление в форме Q-детерминанта. Q-детерминант содержит все реализации алгоритма, в том числе Q-эффективную. Q-эффективная реализация полностью использует ресурс параллелизма алгоритма. С формальной точки зрения она является максимально быстрой реализацией алгоритма. Концепция Q-детерминанта использовалась для исследования ресурса параллелизма алгоритмов [2, 3]. Цель данного исследования – описание подхода, с помощью которого для любого численного алгоритма можно разработать программу, использующую ресурс параллелизма алгоритма полностью.

Исследование параллельной структуры алгоритмов и программ с целью их реализации на параллельных вычислительных системах является важным и развитым. Основа исследования описана в [4, 5]. Для описания параллельных алгоритмов используется представление с помощью графов. В настоящее время создается Интернет-энциклопедия AlgoWiki [6]. В энциклопедии описываются свойства, особенности, статические и динамические характеристики алгоритмов. В докладе [7] представлено современное

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

состояние исследований в области распараллеливания алгоритмов и их реализации на параллельных вычислительных системах. Доклад содержит формулировку некоторых нерешенных проблем. Вот некоторые из них. Как изобразить потенциально бесконечный граф? Как изобразить потенциально многомерный граф? Как показать зависимость структуры графа от размера задачи? Как выразить имеющийся параллелизм и показать возможный способ параллельного исполнения? На эти вопросы отвечает концепция Q-детерминанта. Данное исследование содержит один из вариантов ответа на последний вопрос.

Концепция Q-детерминанта

Пусть α алгоритм для решения алгоритмической проблемы $\bar{y} = F(N, B)$, где $N = \{n_1, \ldots, n_k\}$ – множество параметров размерности проблемы, N может быть пустым, B – множество входных данных (счетное множество переменных), $\bar{y} = (y_1, \ldots, y_m)$ – множество выходных данных, m является вычислимой функцией параметров N. $\bar{N}$ – вектор $(\bar{n}_1, \ldots, \bar{n}_k)$, где $\bar{n}_i$ – некоторое заданное значение параметра n_i $(1 \le i \le k)$. $\{\bar{N}\}$ – множество всевозможных векторов $\bar{N}$. Q – множество операций, используемых алгоритмом α. Любое однозначное отображение $w: \{\bar{N}\} \to V$, где V – множество всех выражений над B и Q, называется безусловным Q-термом. Если при любом $\bar{N} \in \{\bar{N}\}$ и любой интерпретации переменных B $w(\bar{N})$ принимает значение логического типа, то w называется безусловным логическим Q-термом. Пусть $u_1, \ldots, u_l$ – безусловные логические Q-термы, $w_1, \ldots, w_l$ – безусловные Q-термы. Множество пар (u_i, w_i), где $i = 1, \ldots, l$, обозначается $(\bar{u}, \bar{w}) = \{(u_i, w_i)\}_{i=1,\ldots,l}$ и называется условным Q-термом длины l. Счетное множество пар безусловных Q-термов $(\bar{u}, \bar{w}) = \{(u_i, w_i)\}_{i=1,2,\ldots}$ называется условным бесконечным Q-термом, если $\{(u_i, w_i)\}_{i=1,\ldots,l}$ является условным Q-термом для любого $l < \infty$. Если не имеет значения, является Q-терм безусловным, условным или условным бесконечным, то его можно называть Q-термом.

Под вычислением безусловного Q-терма w при интерпретации B следует понимать вычисление выражения $w(\bar{N})$ при некотором $\bar{N} \in \{\bar{N}\}$. Для вычисления при заданной интерпретации B и некотором $\bar{N} \in \{\bar{N}\}$ условного Q-терма $(\bar{u}, \bar{w}) = \{(u_i, w_i)\}_{i=1,\ldots,l}$ необходимо, вычисляя выражения $u_i(\bar{N}), w_i(\bar{N})$, найти $u_{i_0}(\bar{N}), w_{i_0}(\bar{N})$ такие, что $u_{i_0}(\bar{N})$ принимает значение *true*, а значение $w_{i_0}(\bar{N})$ определено. В качестве значения $(\bar{u}, \bar{w})$ нужно взять $w_{i_0}(\bar{N})$. Если установлено, что выражений $u_{i_0}(\bar{N}), w_{i_0}(\bar{N})$ не существует, то значение $(\bar{u}, \bar{w})$ для данной интерпретации B и данного $\bar{N}$ не определено.

Предположим, что I_1, I_2, I_3 подмножества множества $I = (1,\ldots,m)$ такие, что:

1) $I_1 \bigcup I_2 \bigcup I3 = I$;

2) $I_i \bigcap I_j = \varnothing$ $(i \neq j; i, j = 1,2,3)$;

3) одно или два из множеств I_i $(i = 1,2,3)$ могут быть пустыми.

Множество Q-термов $\{f_i\}_{i \in I}$ удовлетворяет условиям:

1) f_{i_1} $(i_1 \in I_1)$ – безусловный Q-терм, $f_{i_1} = w^{i_1}$;

2) f_{i_2} $(i_2 \in I_2)$ – условный Q-терм, $f_{i_2} = \{(u_j^{i_2}, w_j^{i_2})\}_{j=1,\ldots,l_{i_2}}$, l_{i_2} является вычислимой функцией параметров N;

3) f_{i_3} $(i_3 \in I_3)$ – условный бесконечный Q-терм, $f_{i_3} = \{(u_j^{i_3}, w_j^{i_3})\}_{j=1,2,\ldots}$.

Если алгоритм α состоит в том, что для определения y_i $(i \in I)$ требуется вычислить Q-терм f_i, то множество Q-термов y_i $(i \in I)$ называется Q-детерминантом алгоритма α, а представление алгоритма в виде $y_i = f_i$ $(i \in I)$ представлением в форме Q-детерминанта.

Реализацией алгоритма α, представленного в форме Q-детерминанта $y_i = f_i$ $(i \in I)$, называется вычисление Q-термов f_i $(i \in I)$ при заданной интерпретации B и некотором $\overline{N} \in \{\overline{N}\}$. Если реализация такова, что выражения $W(\overline{N}) = \{w^{i_1}(\overline{N})$ $(i_1 \in I_1); u_j^{i_2}(\overline{N}), w_j^{i_2}(\overline{N})$ $(i_2 \in I_2, j = 1,\ldots,l_{i_2}); u_j^{i_3}(\overline{N}), w_j^{i_3}(\overline{N})$ $(i_3 \in I_3, j = 1,2,\ldots)\}$ вычисляются одновременно (параллельно) и при вычислении каждого из выражений операции выполняются по мере их готовности, то реализация называется Q-эффективной. Q-эффективная реализация алгоритма с формальной точки зрения является максимально быстрой. Реализация алгоритма α называется выполнимой, если одновременно необходимо выполнять конечное число операций. Существуют алгоритмы, для которых Q-эффективная реализация не является выполнимой.

Разработка программы для Q-эффективной реализации алгоритма

Подход к разработке программы, выполняющей Q-эффективную реализацию алгоритма, основан на следующих утверждениях:

- Q-детерминант можно построить для любого численного алгоритма;
- Q-детерминант позволяет описать Q-эффективную реализацию алгоритма;
- если Q-эффективная реализация алгоритма является выполнимой, то можно разработать программу для ее выполнения.

Процесс разработки программы состоит из следующих этапов:

1) построение Q-детерминанта алгоритма;

2) описание Q-эффективной реализация алгоритма;

3) если Q-эффективная реализация выполнима, то для нее разрабатывается программа.

Разработанную программу будем называть Q-эффективной, а процесс ее разработки Q-эффективным программированием. Q-эффективная про-

грамма полностью использует ресурс параллелизма алгоритма, так как выполняет его Q-эффективную реализацию. В связи с этим она не допускает дальнейшего распараллеливания.

На этапе 3 для разработки программы должны использоваться средства параллельного программирования. При использовании распределенной памяти исследования ограничиваются вычислениями по принципу «Мастер – Рабочие», который часто применяется на кластерных вычислительных системах. Для дальнейшего изложения вычислительный узел «Мастер» обозначим M, а узел «Рабочий» – P.

Далее будет продемонстрировано применение описанного подхода к некоторым алгоритмам, Q-детерминанты которых состоят из Q-термов различного типа.

Алгоритм умножения матриц

Этап 1. Рассмотрим алгоритм умножения матриц $A = \lfloor a_{ij} \rfloor_{i=1,\ldots,n;\, j=1,\ldots,k}$ и $B = \lfloor b_{ij} \rfloor_{i=1,\ldots,k;\, j=1,\ldots,m}$. Результатом является матрица $C = \lfloor c_{ij} \rfloor_{i=1,\ldots,n;\, j=1,\ldots,m}$, где $c_{ij} = \sum_{s=1}^{k} a_{is} b_{sj}$. Алгоритм представлен в форме Q-детерминанта, состоящего из nm безусловных Q-термов.

Этап 2. Q-эффективная реализация заключается в том, что Q-термы $\sum_{s=1}^{k} a_{is} b_{sj} (i = 1,\ldots,n;\, j = 1,\ldots,m)$ вычисляются одновременно, при этом операции выполняются по мере их готовности к выполнению. Сначала к выполнению готовы все операции умножения, поэтому они должны быть выполнены одновременно. В результате будут получены nm цепочек, образованных с помощью операции сложения. Цепочки должны вычисляться одновременно, при этом при вычислении каждой из них должна использоваться схема сдваивания. Q-эффективная реализация алгоритма умножения матриц выполнима.

Этап 3. При использовании распределенной памяти каждый Q-терм $\sum_{s=1}^{k} a_{is} b_{sj} (i = 1,\ldots,n;\, j = 1,\ldots,m)$ вычисляется на своем вычислительном узле P. Если количество узлов P меньше количества Q-термов, то на одном узле P может вычисляться несколько Q-термов. Результат вычисления каждого из Q-термов передается на узел M.

Метод Гаусса-Жордана для решения систем линейных уравнений

Этап 1. Метод Гаусса-Жордана для решения систем линейных уравнений $A\bar{x} = \bar{b}$ можно применять для любых размерностей. Для простоты будем считать, что $A = \lfloor a_{ij} \rfloor_{i,\, j=1,\ldots,n}$ – матрица размерности $n \times n$ с ненулевым определителем. $\bar{x} = (x_1,\ldots,x_n)^T$, $\bar{b} = (a_{1,n+1},\ldots,a_{n,n+1})^T$ – векторы-столбцы, $\bar{A} = \lfloor a_{ij} \rfloor_{i,=1,\ldots,n;\, j=1,\ldots,n+1}$ – расширенная матрица системы. Построим Q-детерминант метода Гаусса-Жордана. Метод Гаусса-Жордана состоит из n шагов.

Шаг 1. В качестве ведущего элемента выберем a_{11}, если $a_{11} \neq 0$, иначе $a_{1j_1}(j_1 > 1)$, если $a_{1j} = 0$ для $j < j_1 \leq n$ и $a_{1j_1} \neq 0$. Получим расширенную матрицу $\overline{A}^{j_1} = \left[a_{ij}^{j_1}\right]$, элементы которой вычисляются по формулам

$$a_{1j}^{j_1} = \frac{a_{1j}}{a_{1j_1}},\ a_{ij}^{j_1} = a_{ij} - \frac{a_{1j}}{a_{1j_1}} a_{ij_1} (i = 2,\ldots,n;\ j = 1,\ldots,n+1).$$

Шаг k $(2 \leq k \leq n)$. После шага $(k-1)$ будет получена расширенная матрица $\overline{A}^{j_1\ldots j_{k-1}}$. В качестве ведущего элемента выберем $a_{k1}^{j_1\ldots j_{k-1}}$, если $a_{k1}^{j_1\ldots j_{k-1}} \neq 0$, иначе $a_{kj_k}^{j_1\ldots j_{k-1}}(j_k > 1)$, если $a_{kj}^{j_1\ldots j_{k-1}} = 0$ для $j < j_k \leq n$ и $a_{kj_k}^{j_1\ldots j_{k-1}} \neq 0$. Получим расширенную матрицу $\overline{A}^{j_1\ldots j_k} = \left[a_{ij}^{j\ \ldots j_k}\right]$, элементы которой вычисляются по формулам $a_{kj}^{j_1\ldots j_k} = \frac{a_{kj}^{j_1\ldots j_{k-1}}}{a_{kj_k}^{j_1\ldots j_{k-1}}}$, $a_{ij}^{j_1\ldots j_k} = a_{ij}^{j_1\ldots j_{k-1}} - \frac{a_{kj}^{j_1\ldots j_{k-1}}}{a_{kj_k}^{j_1\ldots j_{k-1}}} a_{ij_k}^{j_1\ldots j_{k-1}}$ $(i = 1,\ldots,n; i \neq k;\ j = 1,\ldots,n+1)$.

После шага n имеем систему уравнений $A^{j_1\ldots j_n}\overline{x} = \overline{b}^{j_1\ldots j_n}$, где $A^{j_1\ldots j_n} = \left[a_{ij}^{j_1\ldots j_n}\right]_{i=1,..,n;\ j=1,..,n}$, $\overline{b}^{j_1\ldots j_n} = (a_{1,n+1}^{j_1\ldots j_n},\ldots,a_{n,n+1}^{j_1\ldots j_n})^T$.

Введем обозначения: $L_{j_1} = \bigwedge_{j=1}^{j_1-1}(a_{1j} = 0)$, если $j_1 \neq 1$, $L_{j_1} = true$, если $j_1 = 1$, $L_{j_l} = \bigwedge_{j=1}^{j_l-1}(a_{lj}^{j_1\ldots j_{l-1}} = 0)$, если $j_l \neq 1$, $L_{j_l} = true$, если $j_l = 1$ $(l = 2,...,n)$. В зависимости от значений элементов матрицы A возможны $n!$ способов выбора $(j_1,\ldots,j_n)$, то есть $n!$ перестановок элементов $(1,\ldots,n)$. Занумеруем все перестановки от 1 до $n!$. Пусть i – номер перестановки $(j_1,\ldots,j_n)$. Тогда $w_i^{j_l} = a_{l,n+1}^{j_1\ldots j_n}$ $(l = 1,\ldots,n)$, $u_i = L_{j_1} \wedge (a_{1j_1} \neq 0) \wedge (\bigwedge_{l=2}^{n}(L_{j_l} \wedge (a_{lj_l}^{j_1\ldots j_{l-1}} \neq 0)))$ являются безусловными Q-термами.

Q-детерминант метода Гаусса-Жордана состоит из n условных Q-термов, а представление в форме Q-детерминанта имеет вид $x_j = \{(u_1, w_1^j),\ldots,(u_{n!}, w_{n!}^j)\}$ $(j = 1,\ldots,n)$.

Этап 2. Опишем Q-эффективную реализацию метода Гаусса-Жордана. В соответствии с определением Q-эффективной реализации все безусловные Q-термы $\{u_i, w_i^j\}(i = 1,...,n!;\ j = 1,...,n)$ должны вычисляться одновременно. Поэтому одновременно должны выполняться два вычислительных процесса: параллельное вычисление матриц $\overline{A}^{j_1}, \overline{A}^{j_1 j_2},..., \overline{A}^{j_1 j_2\ldots j_n}$ для любых возможных значений $j_1, j_2,..., j_n$ и параллельное вычисление Q-термов $u_i(i = 1,...,n!)$. При вычислении Q-термов u_i последовательно определяются ведущие элементы матриц для каждого шага алгоритма. Вычисление матриц $\overline{A}^{j_1}, \overline{A}^{j_1 j_2},..., \overline{A}^{j_1 j_2\ldots j_n}$, которые не соответствуют ведущим элементам, прекращается. Если вычисление Q-термов u_i будет идти быстрее, чем матриц $\overline{A}^{j_1}, \overline{A}^{j_1 j_2},..., \overline{A}^{j_1 j_2\ldots j_n}$, то порядок вычислений будет следующим. Первый цикл

состоит в том, что одновременно начинают вычисляться матрицы $\overline{A}^{j_1}$ и Q-термы $u_i(i=1,...,n!)$. Вычисление u_i начинается с подвыражений $L_{j_1} \wedge (a_{1j_1} \neq 0)(j_1 = 1,...,n)$, так как только их операции готовы к выполнению. Только одно из этих подвыражений будет иметь значение *true*. Соответствующее ему значение j_1 обозначим через r_1. Далее прекращается вычисление $\overline{A}^{j_1}$ и u_i, для которых $j_1 \neq r_1$. Во втором цикле одновременно вычисляются матрицы $\overline{A}^{r_1 j_2}(j_2 = 1,...,n; j_2 \neq r_1)$ и Q-термы u_i, для которых $j_1 = r_1$. При вычислении u_i нужно вычислять только подвыражения $L_{j_2} \wedge (a_{2j_2}^{r_1} \neq 0)(j_2 = 1,...,n; j_2 \neq r_1)$, так как только их операции готовы к выполнению. Только одно из этих подвыражений будет иметь значение *true*. Соответствующее ему значение j_2 обозначим через r_2. Далее прекращается вычисление $\overline{A}^{r_1 j_2}(j_2 = 1,...,n; j_2 \neq r_1)$ и $u_i(i=1,...,n!)$, для которых $j_2 \neq r_2$. Следующие $n-3$ цикла вычислений выполняются аналогично. В заключение нужно вычислить единственную матрицу $\overline{A}^{r_1...r_{n-1}j_n}(j_n \neq r_1, r_2, ..., r_{n-1})$, так как j_n имеет единственное значение. Обозначим это значение через r_n. В результате получим решение системы линейных уравнений $x_{r_j} = a_{j,n+1}^{r_1...r_n}(j=1,...,n)$. Q-эффективная реализация метода Гаусса-Жордана выполнима.

Этап 3. При использовании распределенной памяти каждая матрица $\overline{A}^{j_1}(j_1 = 1,...,n)$ и соответствующий ей Q-терм $u_i(i=1,...,n!)$ должны вычисляться на своем узле P. Если количество узлов P меньше n, то они должны выполнять вычисления для нескольких значений j_1. Узлы P получают от узла M информацию для вычисления матриц $\overline{A}^{j_1}$ и Q-термов u_i. Результаты вычисления r_1 и $\overline{A}^{r_1}$ передаются на узел M. Каждая матрица $\overline{A}^{r_1 j_2}(j_2 = 1,...,n; j_2 \neq r_1)$ и соответствующий ей Q-терм u_i должны вычисляться на своем узле P. Узлы P получают от узла M информацию для вычисления матриц $\overline{A}^{r_1 j_2}(j_2 = 1,...,n; j_2 \neq r_1)$ и Q-термов $u_i(i=1,...,n!)$. Результаты вычисления r_2 и $\overline{A}^{r_1 r_2}$ передаются на узел M. Следующие $n-3$ цикла вычислений выполняются аналогично. Последний цикл состоит в вычислении единственной матрицы $\overline{A}^{r_1 r_2 ... r_n}$, поэтому выполняется на узле M.

Метод Якоби для решения систем линейных уравнений

Этап 1. Построим Q-детерминант метода Якоби для решения системы линейных уравнений $A\overline{x} = \overline{b}$, где $A = [a_{ij}]_{i,j=1,...,n}$, $a_{ii} \neq 0 (i = 1,...,n)$, $\overline{x} = (x_1,...,x_n)^T$, $\overline{b} = (a_{1,n+1},...,a_{n,n+1})^T$. Введем обозначения $c_{ij} = -\dfrac{a_{ij}}{a_{ii}}$, $d_i = \dfrac{b_i}{a_{ii}} (i, j = 1,...,n)$. Пусть $\overline{x}^0$ начальное приближение. Тогда итерационный процесс можно записать так: $x_i^{k+1} = \sum_{j=1,...,n; j \neq i} c_{ij} x_j^k + d_i (i = 1,...,n; k = 0,1,...)$. Критерием окончания итерацион-

ного процесса является $\left\|\bar{x}^{k+1}-\bar{x}^{k}\right\|<\varepsilon$. Здесь ε обозначает точность вычислений. Q-детерминант метода Якоби состоит из n условных бесконечных Q-термов. Представление метода Якоби в форме Q-детерминанта имеет вид $x_i=\{(\left\|\bar{x}^{1}-\bar{x}^{0}\right\|<\varepsilon, x_i^1),...,(\left\|\bar{x}^{k}-\bar{x}^{k-1}\right\|<\varepsilon, x_i^k),...\}(i=1,...,n)$.

Этап 2. Для упрощения описания Q-эффективной реализации введем обозначения $u^l=\left\|\bar{x}^{l}-\bar{x}^{l-1}\right\|<\varepsilon(l=1,2,...)$. Тогда Q-детерминант имеет вид $x_i=\{(u^1,x_i^1),(u^2,x_i^2),...,(u^k,x_i^k),...\}(i=1,...,n)$. В соответствии с определением Q-эффективной реализации все безусловные Q-термы $\{u^l,x_i^l\}(i=1,...,n;l=1,2,...)$ должны вычисляться одновременно. Сначала должны быть вычислены одновременно Q-термы $x_i^1(i=1,...,n)$, так как их операции готовы к выполнению. Затем вычисляются одновременно Q-термы $u^1,x_i^2(i=1,...,n)$. Если u^1 имеет значение *true*, то вычисления заканчиваются, а решением системы линейных уравнений является $x_i=x_i^1(i=1,...,n)$. Если вычисление будет продолжаться, то Q-термы $u^k,x_i^{k+1}(i=1,...,n)$ будут вычисляться одновременно при любом значении $k\geq 2$. Если значение u^k *true*, то вычисления заканчиваются, а решением системы линейных уравнений является $x_i=x_i^k(i=1,...,n)$. Q-эффективная реализация метода Якоби выполнима.

Этап 3. При использовании распределенной памяти каждая компонента вектора $\bar{x}^{k}(k=1,2,...)$ вычисляется на своем вычислительном узле P. Если количество узлов P меньше n, то они должны выполнять вычисления для нескольких компонент вектора $\bar{x}^{k}(k=1,2,...)$. Сначала узел M посылает на узлы P необходимую информацию для вычисления Q-термов $x_i^1(i=1,...,n)$. Результаты вычисления передаются на узел M. Узел M посылает на узлы P значения $x_i^1(i=1,...,n)$. Узел M вычисляет Q-терм $\left\|\bar{x}^{1}-\bar{x}^{0}\right\|<\varepsilon$. Вместе с этим на узлах P вычисляются $x_i^2(i=1,...,n)$ одновременно. Следующие итерации выполняются аналогично.

В настоящее время для рассмотренных алгоритмов разработаны Q-эффективные программы. Разработку программ выполнили студенты Южно-Уральского государственного университета Н. Валькевич, Ю. Лаптева и Д. Тарасов. Для создания Q-эффективных программ был использован язык программирования *C*. Для общей памяти применялась технология OpenMP, для распределенной MPI и OpenMP. Исследование проводилось на суперкомпьютере «Торнадо ЮУрГУ».

Заключение

С помощью предлагаемого подхода к проектированию параллельных программ на основе концепции Q-детерминанта можно разработать Q-эффективную программу для любого численного алгоритма. Q-эффективная программа использует ресурс параллелизма алгоритма

полностью, так как выполняет его Q-эффективную реализацию, поэтому дальнейшее ее распараллеливание невозможно.

Для любого численного алгоритма можно разработать много Q-эффективных программ, так как можно использовать различные языки программирования, различные технологии параллельного программирования, кроме того, Q-эффективная программа ориентирована на архитектурные особенности параллельных вычислительных систем, для которых разрабатывается. Q-эффективные программы могут отличаться по скорости выполнения.

Библиографический список

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

2. Свирихин, Д.И. Определение ресурса параллелизма алгоритма и его эффективного использования для конечного числа процессоров / Д.И. Свирихин // Научный сервис в сети Интернет: поиск новых решений: труды Международной суперкомпьютерной конференции (17–22 сентября 2012 г., г. Новороссийск). – М.: Изд. МГУ, 2012. – С. 257–260.

3. Aleeva, V.N. Software System for Maximal Parallelization of Algorithms on the Base of the Conception of Q-determinant / V.N. Aleeva, I.S. Sharabura, D.E. Suleymanov // In: Malyshkin, V. (ed.) PaCT 2015. LNCS, Vol. 9251. – Switzerland.: Springer, 2015. – P. 3–9.

4. Voevodin, V.V. The V-Ray technology of optimizing programs to parallel computers / V.V. Voevodin, V.V. Voevodin // In: Vulkov, L.G., Yalamov, P., Wasniewski, J. (eds.) WNAA 1996. LNCS, vol. 1196. – Heidelberg.: Springer, 1997. – Pp. 546–556.

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

6. Антонов, А.С. Первая версия Открытой энциклопедии свойств алгоритмов / А.С. Антонов, Вад. В. Воеводин, Вл.В. Воеводин, А.М. Теплов, А.В. Фролов // Сборник «Параллельные вычислительные технологии (ПаВТ’2015): труды международной научной конференции» (31 марта – 2 апреля 2015 г., г. Екатеринбург). – Челябинск: Издательский центр ЮУрГУ, 2015. – С. 31–42.

7. Воеводин, Вл.В. Параллельные алгоритмы под микроскопом / Вл.В. Воеводин // Международная конференция «Параллельные вычислительные технологии (ПаВТ’2016)» (28 марта – 1 апреля 2016 г., г. Архангельск). – URL: http://omega.sp.susu.ru/books/conference/PaVT2016/talks/Voevodin.pdf.

К содержанию