Q-эффективная реализация численных алгоритмов[1]

В.Н. Алеева

Кафедра системного программирования
Южно-Уральский государственный университетет (НИУ)

Доклад на заседании научного семинара по информационным технологиям под руководством профессора Л.Б. Соколинского

Челябинск, 2017

[1]Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-00865 а.

Введение

- История развития параллельных вычислительных систем насчитывает десятки лет, однако эффективность реализации на них алгоритмов остается низкой.
- Один из путей решения проблемы — использование ресурса параллелизма алгоритмов полностью.
- Выявить ресурс параллелизма численных алгоритмов позволяет концепция Q-детерминанта.
- Основой концепции является универсальное описание численных алгоритмов — представление в форме Q-детерминанта.
- Q-детерминант содержит все реализации алгоритма, в том числе Q-эффективную.
- Q-эффективная реализация использует ресурс параллелизма алгоритма полностью.

Исследования на основе концепции Q-детерминанта в настоящее время проводятся в двух направлениях:

1. Автоматизированное исследование ресурса параллелизма численных алгоритмов.
2. Разработка подхода, позволяющего для любого численного алгоритма конструировать программу, использующую полностью ресурс параллелизма алгоритма, а также проводить анализ динамических характеристик полученной программы и ее масштабируемости.

Результаты, представленные в докладе, относятся ко второму направлению исследований.

Цель исследования, о котором пойдет речь в докладе —

описание метода, с помощью которого для любого численного алгоритма можно разработать программу, использующую ресурс параллелизма алгоритма полностью.

- Как известно, направление, связанное с исследованием параллельной структуры алгоритмов и программ с целью их реализации на параллельных вычислительных системах, является очень важным и развитым.
- Одним из основателей этого направления является Валентин Васильевич Воеводин.
- Его работы в этой области стали классическими. В них для описания параллельных алгоритмов используется представление с помощью графов.
- В настоящее время создается Интернет-энциклопедия AlgoWiki.
- В энциклопедии описываются свойства, особенности, статические и динамические характеристики алгоритмов.

В докладе Вл.В. Воеводина “Параллельные алгоритмы под микроскопом” на международной конференции ПаВТ’2016 представлено современное 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.
В докладе речь пойдет об одном из вариантов ответа на последний вопрос.

Концепция Q-детерминанта

$\mathcal{A}$ — 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 $(1 \leqslant i \leqslant k)$.
$\{\bar{N}\}$ — множество всевозможных векторов $\bar{N}$.
Q — множество операций, используемых алгоритмом $\mathcal{A}$.

Определение Q-терма

1. Любое однозначное отображение $w : \{\bar{N}\} \to V$, где V — множество всех выражений над B и Q, называется *безусловным Q-термом*.
2. 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*.
3. Пусть $u_1, \ldots, u_l$ — безусловные логические Q-термы, $w_1, \ldots, w_l$ — безусловные Q-термы.
   Множество l 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*.
4. 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*.

Вычисление Q-терма

1. *Под вычислением безусловного Q-терма w при заданной интерпретации B* и некотором $\bar{N} \in \{\bar{N}\}$ следует понимать вычисление выражения $w(\bar{N})$.
2. Для *вычисления при заданной интерпретации B и некотором $\bar{N} \in \{\bar{N}\}$ условного Q-терма* $(\bar{u}, \bar{w}) = \{(u_i, w_i)\}_{i=1,\dots,l}$ необходимо, вычисляя выражения $u_i(\bar{N}), w_i(\bar{N})$ $(i = 1, \dots, l)$, 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 $w_{i_0}(\bar{N})$.
3. Для *вычисления при заданной интерпретации B и некотором $\bar{N} \in \{\bar{N}\}$ условного бесконечного Q-терма* $(\bar{u}, \bar{w}) = \{(u_i, w_i)\}_{i=1,2,\dots}$ необходимо, вычисляя выражения $u_i(\bar{N}), w_i(\bar{N})$ $(i = 1, 2, \dots)$, 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 $w_{i_0}(\bar{N})$.

Определение Q-детерминанта алгоритма

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

1) $I_1 \cup I_2 \cup I_3 = I$;
2) $I_i \cap I_j = \varnothing$ $(i \neq j; i, j = 1, 2, 3)$;
3) Одно или два из множеств I_i $(i = 1, 2, 3)$ могут быть пустыми.

Предположим, что 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, $f_{i_2} = \left\{ \left(u_j^{i_2}, w_j^{i_2} \right) \right\}_{j=1,\ldots,l_{i_2}}$, l_{i_2} является вычислимой функцией параметров N;
3) f_{i_3} $(i_3 \in I_3)$ — условный бесконечный Q-терм, $f_{i_3} = \left\{ \left(u_j^{i_3}, w_j^{i_3} \right) \right\}_{j=1,2,\ldots}$.

Если алгоритм $\mathcal{A}$ состоит в том, что для определения y_i $(i \in I)$ требуется вычислить Q-терм f_i, то множество Q-термов f_i $(i \in I)$ называется *Q-детерминантом алгоритма $\mathcal{A}$*, а представление алгоритма в виде

$$y_i = f_i \ (i \in I)$$

представлением в форме Q-детерминанта.

Определение Q-эффективной реализации алгоритма

Реализацией алгоритма $\mathcal{A}$, 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 $\bar{N} \in \{\bar{N}\}$.
Если две или более операций реализации выполняются одновременно, то такую реализацию будем называть параллельной.
Если реализация такова, что выражения

$$W(\bar{N}) = \left\{ w^{i_1}(\bar{N})(i_1 \in I_1); u_j^{i_2}(\bar{N}), w_j^{i_2}(\bar{N})(i_2 \in I_2, j = 1, \ldots, l_{i_2}); \right.$$
$$\left. u_j^{i_3}(\bar{N}), w_j^{i_3}(\bar{N})(i_3 \in I_3, j = 1, 2, \ldots) \right\}$$

вычисляются одновременно (параллельно) и при вычислении каждого из выражений $W(\bar{N})$ операции выполняются по мере их готовности, то реализация называется *Q-эффективной*.

Свойства Q-эффективной реализации алгоритма

- Очевидно, что Q-эффективная реализация алгоритма с формальной точки зрения является максимально быстрой реализацией алгоритма.
- Очевидно также, что если алгоритм допускает распараллеливание, то его Q-эффективная реализация полностью использует ресурс параллелизма алгоритма, поэтому является максимально параллельной реализаций алгоритма.
- Реализация алгоритма называется *выполнимой*, если одновременно необходимо выполнять конечное число операций.
  Существуют алгоритмы, для которых Q-эффективная реализация не является выполнимой.

Метод проектирования параллельной программы для Q-эффективной реализации алгоритма

Метод основан на следующих утверждениях:

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

Метод проектирования параллельной программы для Q-эффективной реализации алгоритма

Метод состоит из следующих этапов:

этап 1. построение Q-детерминанта алгоритма;

этап 2. описание плана выполнения Q-эффективной реализации алгоритма;

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

Метод проектирования параллельной программы для Q-эффективной реализации алгоритма

- Полученную с помощью описанного метода программу будем называть Q-эффективной, а процесс ее создания Q-эффективным программированием.
- Q-эффективная программа полностью использует ресурс параллелизма алгоритма, так как выполняет его Q-эффективную реализацию.
- В связи с этим Q-эффективная программа не допускает дальнейшего распараллеливания.
- Далее будет продемонстрировано применение описанного метода к некоторым алгоритмам, Q-детерминанты которых состоят из Q-термов различного типа.
- Сначала будут описаны первые два этапа метода.

Алгоритм умножения матриц

Этап 1.

Рассмотрим алгоритм умножения матриц

$$A = [a_{ij}]_{i=1,\ldots,n;j=1,\ldots,k} \text{ и } B = [b_{ij}]_{i=1,\ldots,k;j=1,\ldots,m}.$$

Результатом является матрица $C = [c_{ij}]_{i=1,\ldots,n;j=1,\ldots,m}$, 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 все 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 вычислении каждой из них должна использоваться схема сдваивания.
- Q-эффективная реализация алгоритма умножения матриц выполнима.

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. В нашем случае будем считать, что $A = [a_{ij}]$ — 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, $\bar{A}$ 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 $j < j_1 \leqslant n$ и $a_{1j_1} \neq 0$.
Получим расширенную матрицу $\bar{A}^{j_1} = [a_{ij}^{j_1}]$, 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 $(2 \leqslant k \leqslant n)$. После шага $(k-1)$ имеем расширенную матрицу $\bar{A}^{j_1 \ldots j_{k-1}}$. 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 $j < j_k \leqslant n$ и $a_{kj_k}^{j_1 \ldots j_{k-1}} \neq 0$.
Получим расширенную матрицу $\bar{A}^{j_1 \ldots j_k} = [a_{ij}^{j_1 \ldots j_k}]_{i=1,\ldots,n; j=1,\ldots,n+1}$, 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 получаем систему уравнений $A^{j_1 \ldots j_n} \bar{x} = \bar{b}^{j_1 \ldots j_n}$, где

$$A^{j_1 \ldots j_n} = [a_{ij}^{j_1 \ldots j_n}]_{i=1,\ldots,n; j=1,\ldots,n}, \; \bar{b}^{j_1 \ldots j_n} = (a_{1,n+1}^{j_1 \ldots j_n}, \ldots, a_{n,n+1}^{j_1 \ldots j_n})^T.$$

Введем обозначения:

$$L_{j_1} = \bigwedge_{j=1}^{j_1-1}(a_{1j} = 0), \text{ если } j_1 \neq 1, \ L_{j_1} = \texttt{true}, \text{ если } j_1 = 1,$$

$$L_{j_l} = \bigwedge_{j=1}^{j_l-1}(a_{lj}^{j_1 \dots j_{l-1}} = 0) \text{ если } j_l \neq 1, L_{j_l} = \texttt{true}, \text{ если } j_l = 1 \ (l = 2, \dots, n).$$

Перестановки элементов $(1, \dots, n)$ могут быть пронумерованы.
Пусть i номер перестановки $(j_1, \dots, j_n)$. Тогда

$$w_i^{j_l} = a_{l,n+1}^{j_1 \dots j_n}(l = 1, \dots, n), u_i = L_{j_1} \wedge (a_{1j_1} \neq 0) \wedge \left(\bigwedge_{l=2}^{n} \left(L_{j_l} \wedge \left(a_{lj_l}^{j_1 \dots j_{l-1}} \neq 0 \right) \right) \right)$$

являются безусловными Q-термами.
Q-детерминант метода Гаусса-Жордана состоит из n условных Q-термов, а представление в форме Q-детерминанта имеет вид

$$x_j = \{(u_1, w_1^j), \dots, (u_{n!}, w_{n!}^j)\}(j = 1, \dots, n).$$

Этап 2.

- Опишем план выполнения Q-эффективной реализации метода Гаусса-Жордана.
- В соответствии с определением Q-эффективной реализации все безусловные Q-термы $\{u_i, w_i^j\}(i{=}1, \ldots, n!; j{=}1, \ldots, n)$ должны вычисляться одновременно, при этом операции должны выполняться по мере их готовности к выполнению.
- Эти требования приводят к тому, что одновременно должно выполняться:

  параллельное вычисление матриц $\bar{A}^{j_1}, \bar{A}^{j_1 j_2}, \ldots, \bar{A}^{j_1 j_2 \ldots j_n}$ для любых возможных значений $j_1, j_2, \ldots, j_n$

  и параллельное вычисление Q-термов $u_i(i{=}1, \ldots, n!)$.
- При вычислении Q-термов $u_i(i{=}1, \ldots, n!)$ один за другим определяются ведущие элементы матриц для всех шагов алгоритма.
- После определения очередного ведущего элемента вычисление матриц $\bar{A}^{j_1}, \bar{A}^{j_1 j_2}, \ldots, \bar{A}^{j_1 j_2 \ldots j_n}$ и Q-термов u_i, которые не соответствуют ведущему элементу, прекращается.

Вычисление Q-термов $u_i(i = 1, \ldots, n!)$ выполняется быстрее, чем матриц $\bar{A}^{j_1}, \bar{A}^{j_1 j_2}, \ldots, \bar{A}^{j_1 j_2 \cdots j_n}$, т.е. ведущий элемент любого шага k будет определяться раньше, чем будут вычислены матрицы $\bar{A}^{j_1 j_2 \cdots j_k}$, поэтому порядок вычислений следующий:

1. Первый цикл состоит в том, что одновременно начинают вычисляться матрицы $\bar{A}^{j_1}$ и Q-термы $u_i(i = 1, \ldots, n!)$.

   Вычисление $u_i(i{=}1, \ldots, n!)$ начинается с подвыражений $L_{j_1} \wedge (a_{1 j_1} \neq 0)(j_1{=}1, \ldots, n)$, так как только их операции готовы к выполнению.

   Только одно из подвыражений $L_{j_1} \wedge (a_{1 j_1} \neq 0)(j_1{=}1, \ldots, n)$ будет иметь значение `true`.

   Соответствующее ему значение j_1 обозначим через r_1.

   Далее прекращается вычисление $\bar{A}^{j_1}(j_1 = 1, \ldots, n)$ и $u_i(i = 1, \ldots, n!)$, для которых $j_1 \neq r_1$.

2. Во втором цикле одновременно вычисляются матрицы $\bar{A}^{r_1 j_2}(j_2 = 1, \ldots, n; j_2 \neq r_1)$ и Q-термы $u_i(i = 1, \ldots, n!)$, для которых $j_1 = r_1$.
   При вычислении $u_i(i = 1, \ldots, n!)$ нужно вычислять только подвыражения $L_{j_2} \wedge (a^{r_1}_{2j_2} \neq 0)(j_2 = 1, \ldots, n; j_2 \neq r_1)$, так как только их операции готовы к выполнению.
   Только одно подвыражение $L_{j_2} \wedge (a^{r_1}_{2j_2} \neq 0)(j_2 = 1, \ldots, n; j_2 \neq r_1)$ будет иметь значение `true`.
   Соответствующее ему значение j_2 обозначим через r_2.
   Далее прекращается вычисление $\bar{A}^{r_1 j_2}(j_2 = 1, \ldots, n; j_2 \neq r_1)$ и $u_i(i = 1, \ldots, n!)$, для которых $j_2 \neq r_2$.
3. Следующие $n - 3$ цикла вычислений выполняются аналогично.
4. В заключение нужно вычислить единственную матрицу $\bar{A}^{r_1 \ldots r_{n-1} j_n}(j_n \neq r_1, r_2, \ldots, r_{n-1})$, так как j_n имеет единственное значение. Обозначим это значение через r_n.
5. В результате получим решение системы линейных уравнений

$$x_{r_j} = a^{r_1 \ldots r_n}_{j,n+1}(j = 1, \ldots, n).$$

Q-эффективная реализация метода Гаусса-Жордана выполнима.

Метод Якоби для решения систем линейных уравнений

Этап 1.

- Построим Q-детерминант метода Якоби для решения системы линейных уравнений $A\bar{x}=\bar{b}$, где $A=[a_{ij}]_{i,j=1,\dots,n}$, $a_{ii} \neq 0$ $(i=1,\dots,n)$, $\bar{x}=(x_1,\dots,x_n)^T$, $\bar{b}=(a_{1,n+1},\dots,a_{n,n+1})^T$.
- Введем обозначения $c_{ij} = -\frac{a_{ij}}{a_{ii}}$, $d_i = \frac{b_i}{a_{ii}} (i,j=1,\dots,n)$.

  Пусть $\bar{x}^0$ начальное приближение.

  Тогда итерационный процесс можно записать так

$$x_i^{k+1} = \sum_{j=1,\dots,n;j\neq i} c_{ij}x_j^k + d_i \ (i=1,\dots,n;\ k=0,1,\dots).$$

  Критерием окончания итерационного процесса является

$$||\bar{x}^{k+1} - \bar{x}^k|| < \varepsilon.$$

  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

$$x_i=\{(||\bar{x}^1-\bar{x}^0||<\varepsilon, x_i^1),\dots,(||\bar{x}^k-\bar{x}^{k-1}||<\varepsilon, x_i^k),\dots\}(i=1,\dots,n).$$

Этап 2.

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

Этап 3 предлагаемого метода заключается в разработке программы для Q-эффективной реализации алгоритма.

Перед тем, как приступить к разработке программы, нужно определиться, что использовать, например:

1. Какие архитектурные особенности ВС (например, общую память, распределенную память, сопроцессоры, принципы коммутации между узлами, векторизацию и т.д.)?
2. Какой язык программирования и компилятор?
3. Какие средства параллельного программирования (например, OpenMP, MPI, MPI+OpenMP, CUDA, OpenCL и т.д.)?

В рамках данного исследования рассматривается создание Q-эффективных программ для платформ с общей и распределенной памятью.
При использовании общей памяти разработка Q-эффективной программы осуществляется в соответствии с планом выполнения Q-эффективной реализации.

- Опишем особенности разработки Q-эффективных программ для рассмотренных алгоритмов при использовании распределенной памяти, дополнив план выполнения Q-эффективной реализации алгоритма планом распределения вычислений по вычислительным узлам.
- При использовании распределенной памяти данные исследования ограничиваются вычислениями по принципу «Мастер – Рабочие», который часто применяется на кластерных вычислительных системах.
- Для вычисления используется вычислительный узел M (Мастер) и несколько узлов P (Рабочий).
- В соотвествии с этим принципом вычислительный процесс разбивается на несколько шагов:

  Шаг 0. Инициализация.

  Шаг 1. Посылка задания от узла M всем узлам P.

  Шаг 2. Вычисление на каждом узле P без обменов с другими узлами.

  Шаг 3. Посылка результатов от всех узлов P узлу M.

  Шаг 4. Слияние на узле M полученных результатов.

Алгоритм умножения матриц

- 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

- Каждая матрица $\bar{A}^{j_1}(j_1 = 1, \ldots, n)$ и соответствующий ей Q-терм $u_i(i{=}1, \ldots, n!)$ должны вычисляться на своем узле P.

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

Метод Якоби для решения систем линейных уравнений

- Каждая компонента вектора $\bar{x}^k (k = 1, 2, \dots)$ вычисляется на своем вычислительном узле P.
- Если количество узлов P меньше n, то узел P может выполнять вычисления для нескольких компонент вектора $\bar{x}^k (k = 1, 2, \dots)$.
- Сначала узел M передает на узлы P необходимую информацию для вычисления Q-термов $x_i^1 (i = 1, \dots, n)$.

  Результаты вычисления передаются на узел M.
- Узел M передает на узлы P значения $x_i^1 (i = 1, \dots, n)$.
- Узел M вычисляет Q-терм $||\bar{x}^1 - \bar{x}^0|| < \varepsilon$.

  Вместе с этим на узлах P вычисляются $x_i^2 (i = 1, \dots, n)$ одновременно.
- Следующие итерации выполняются аналогично.

- Для рассмотренных алгоритмов были разработаны Q-эффективные программы для общей и распределенной памяти.
- Исследование проведено на суперкомпьютере “Торнадо ЮУрГУ”.
- При разработке программ использовался язык программирования C++.
- Для общей памяти применялась технология OpenMP, для распределенной MPI и OpenMP.
- Разработку Q-эффективных программ выполнили:
  Николай Валькевич (бакалавриат, 4 курс) - алгоритм умножения плотных и разреженных матриц;
  Денис Тарасов (магистрантура, 2 курс) - метод Гаусса-Жордана для решения систем линейных уравнений;
  Юлия Лаптева (бакалавриат, 4 курс) - метод Якоби для решения систем линейных уравнений.
- На основе результатов проведенного исследования вычислительная модель концепции Q-детерминанта будет расширена с целью учета зависимости реализаций алгоритмов, в том числе Q-эффективной, от потерь, возникающих при работе с памятью в процессе выполнения на параллельной вычислительной системе.

Заключение

- С помощью предлагаемого метода проектирования 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 Q-эффективную реализацию, поэтому дальнейшее ее распараллеливание невозможно.
- Для любого численного алгоритма может быть разработано много Q-эффективных программ, так как Q-эффективная программа ориентирована на архитектурные особенности вычислительных систем, для которых разрабатывается, для ее разработки могут использоваться различные языки программирования, различные технологии параллельного программирования и т.д.
- Q-эффективные программы, разработанные для одного и того же алгоритма, могут отличаться по скорости выполнения.
- Результаты данного исследования могут быть применены для повышения эффективности реализации численных алгоритмов на параллельных вычислительных системах.
- Динамические характеристики, масштабируемость Q-эффективных программ являются предметом отдельного исследования.