

Решение проблемы повышения эффективности параллельных вычислений¹

В.Н. Алеева

Южно-Уральский государственный университет (НИУ)

Международная научная конференция
“Алгебра и динамические системы”,
посвященная 110-летию со дня рождения С.Н. Черникова
Нальчик, 2022

¹Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-00865 а “Метод исследования ресурса параллелизма численных алгоритмов и суперкомпьютерный кодизайн на основе концепции Q -детерминанта”.

Актуальность исследования

- ▶ Алгоритмы, как правило, реализуются так, что они не применяют вычислительные ресурсы параллельных вычислительных систем настолько, насколько способны это делать.
- ▶ Эффективность параллельных вычислений можно повысить, если использовать весь внутренний ресурс параллелизма алгоритмов.
- ▶ Проблема исследования внутреннего ресурса параллелизма алгоритмов и его использования для повышения эффективности параллельных вычислений является актуальной, а ее решение имеет научное и практическое значение.

Цель и задачи исследования

Цель исследования — разработка решений для повышения эффективности параллельных вычислений, использующих численные алгоритмы.

Для достижения цели решены следующие задачи.

1. Разработана программная *Q*-система для исследования ресурса параллелизма численных алгоритмов.
2. Разработан метод проектирования эффективных программ, полностью использующих ресурс параллелизма численных алгоритмов.
3. Разработана технология *Q*-эффективного программирования для повышения эффективности реализации численных методов и алгоритмических проблем.

Теоретический фундамент для исследования

- ▶ Теоретическим фундаментом для исследования является концепция Q -детерминанта.
- ▶ Концепция Q -детерминанта — один из подходов к распараллеливанию численных алгоритмов.
- ▶ Концепция основана на универсальном описании численных алгоритмов — представлении в форме Q -детерминанта.

Алгоритмическая проблема

Алгоритмическая проблема может быть представлена в виде

$$\bar{y} = F(N, B),$$

где $N = \{n_1, \dots, n_k\}$ — множество параметров размерности проблемы или пустое множество,

B — множество входных данных,

$\bar{y} = (y_1, \dots, y_m)$ — множество выходных данных, m является вычислимой функцией параметров N .

Далее через \bar{N} будем обозначать вектор $(\bar{n}_1, \dots, \bar{n}_k)$, где \bar{n}_i — некоторое заданное значение параметра n_i ($1 \leq i \leq k$).

Кратко о понятии Q -детерминанта алгоритма

- ▶ Пусть \mathcal{A} — численный алгоритм для решения алгоритмической проблемы,
 Q — множество операций, используемых алгоритмом \mathcal{A} ,
 B — множество входных и $\bar{y} = (y_1, \dots, y_m)$ — множество выходных данных алгоритма \mathcal{A} .
- ▶ Для каждого y_i ($i = 1, \dots, m$) строится Q -терм f_i , который описывает все способы вычисления y_i в зависимости от B в соответствии с алгоритмом \mathcal{A} .
- ▶ Q -детерминант алгоритма \mathcal{A} — это множество Q -термов f_i .
- ▶ Представление алгоритма \mathcal{A} в виде $y_i = f_i$ ($i = 1, \dots, m$) называется представлением в форме Q -детерминанта.

Примеры представления алгоритмов в форме Q -детерминантов

- ▶ Алгоритм умножения плотных матриц сразу представлен в форме Q -детерминанта $c_{ij} = \sum_{s=1}^k a_{is} b_{sj}$ ($i = 1, \dots, n; j = 1, \dots, m$) из nm безусловных Q -термов.
 - ▶ Однако численные алгоритмы чаще не представлены в форме Q -детерминанта, но их представление можно найти.
- 1) Представление в форме Q -детерминанта алгоритма, реализующего метод Гаусса—Жордана решения СЛАУ (n — размер матрицы):

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

Q -детерминант состоит из n условных Q -термов длины $n!$.

- 2) Представление в форме Q -детерминанта алгоритма, реализующего метод Якоби решения СЛАУ (n — размер матрицы, ε — точность вычислений):

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

Q -детерминант состоит из n условных бесконечных Q -термов.

Q-эффективная реализация алгоритма

- ▶ Реализацией алгоритма \mathcal{A} , представленного в форме Q -детерминанта, называется вычисление Q -термов.
- ▶ Если реализация такова, что Q -термы вычисляются одновременно (параллельно) и при вычислении каждого из них операции выполняются по мере их готовности, то реализация называется *Q*-эффективной.
- ▶ Если алгоритм допускает распараллеливание, то его *Q*-эффективная реализация полностью использует ресурс параллелизма алгоритма.
- ▶ Реализация алгоритма называется выполнимой, если при реализации одновременно необходимо выполнять конечное число операций.

Характеристики ресурса параллелизма алгоритма

Пусть $W(\bar{N})$ — выражения, из которых состоят Q -термы Q -детерминанта алгоритма \mathcal{A} ,

$T^{w(\bar{N})}$ — уровень вложенности выражения $w(\bar{N})$.

Введем характеристики:

- 1) $D_{\mathcal{A}}(\bar{N})$ — максимальное число уровней вложенности выражений

$W(\bar{N})$, т.е. $D_{\mathcal{A}}(\bar{N}) = \max_{w(\bar{N}) \in W(\bar{N})} T^{w(\bar{N})}$;

- 2) $P_{\mathcal{A}}(\bar{N})$ — максимальное из значений, представляющих количество операций на каждом из уровней вложенности всех выражений $W(\bar{N})$,

т.е. $P_{\mathcal{A}}(\bar{N}) = \max_{1 \leq r \leq D_{\mathcal{A}}(\bar{N})} \sum_{w(\bar{N}) \in W(\bar{N})} O_r^{w(\bar{N})}$, где $O_r^{w(\bar{N})}$ — количество операций уровня вложенности r выражения $w(\bar{N})$.

$D_{\mathcal{A}}(\bar{N})$ будем называть высотой алгоритма, $P_{\mathcal{A}}(\bar{N})$ — его шириной.

$D_{\mathcal{A}}(\bar{N})$ характеризует время выполнения Q -эффективной реализации алгоритма,

$P_{\mathcal{A}}(\bar{N})$ — количество процессоров, необходимое для ее выполнения.

ЗАДАЧА 1. Программная Q -система для исследования ресурса параллелизма численных алгоритмов

Для автоматизированного исследования ресурса параллелизма численных алгоритмов были разработаны следующие методы.

- 1) Метод построения Q -детерминанта алгоритма на основе его блок-схемы.
- 2) Метод получения Q -эффективной реализации алгоритма с помощью Q -детерминанта.
- 3) Метод вычисления характеристик ресурса параллелизма алгоритма.
- 4) Метод сравнения характеристик ресурса параллелизма двух алгоритмов.

ЗАДАЧА 1. Программная *Q*-система для исследования ресурса параллелизма численных алгоритмов

- ▶ В настоящее время разработана программная система, реализующая данные методы.
- ▶ Она получила название «*Q*-система (*Q*-system)».
- ▶ В режиме просмотра информации *Q*-система доступна всем по адресу

<https://qclient.herokuapp.com>

ЗАДАЧА 1. Программная Q -система для исследования ресурса параллелизма численных алгоритмов

Q -система позволяет:

- 1) оценить характеристики ресурса параллелизма любого численного алгоритма \mathcal{A} —
высоту $D_{\mathcal{A}}(\bar{N})$ и ширину $P_{\mathcal{A}}(\bar{N})$;
- 2) из множества численных алгоритмов, решающих одну и ту же алгоритмическую проблему, выбрать алгоритм с лучшим ресурсом параллелизма.

ЗАДАЧА 1. Программная Q -система для исследования ресурса параллелизма численных алгоритмов

Q -система состоит из двух подсистем:

- 1.** для создания Q -детерминантов алгоритмов;
- 2.** для вычисления ресурса параллелизма алгоритмов.

ЗАДАЧА 1. Программная Q -система для исследования ресурса параллелизма численных алгоритмов

- ▶ Первая подсистема по блок-схеме алгоритма формирует представление алгоритма в форме Q -детерминанта для фиксированных значений параметров размерности \bar{N} и количества итераций алгоритма.
- ▶ Сначала была разработана версия подсистемы для персонального компьютера.
В настоящее время создана версия для суперкомпьютера.
- ▶ Для описания входных и выходных данных подсистемы применяется язык *JSON*.

```

{
  "vertices": [
    {"id": 1, "Type": 0, "Content": "Start"}, {"id": 2, "Type": 4, "Content": "[n,n]"}, {"id": 3, "Type": 4, "Content": "B[n]"}, {"id": 4, "Type": 4, "Content": "X0[n]"}, {"id": 5, "Type": 4, "Content": "I"}, {"id": 6, "Type": 3, "Content": "Iterations"}, {"id": 7, "Type": 3, "Content": "i<1"}, {"id": 8, "Type": 3, "Content": "i=1"}, {"id": 9, "Type": 3, "Content": "i>n"}, {"id": 10, "Type": 2, "Content": "x(i)=x0(i)"}, {"id": 11, "Type": 2, "Content": "i=i+1"}, {"id": 12, "Type": 2, "Content": "i=1"}, {"id": 13, "Type": 3, "Content": "i<=n"}, {"id": 14, "Type": 2, "Content": "newX(i)=B(i)"}, {"id": 15, "Type": 2, "Content": "i<=m"}, {"id": 16, "Type": 3, "Content": "i=m"}, {"id": 17, "Type": 3, "Content": "i=j"}, {"id": 18, "Type": 3, "Content": "j<1"}, {"id": 19, "Type": 2, "Content": "D=A(i,j)*newX(j)"}, {"id": 20, "Type": 2, "Content": "D=A(i,j)*x(j)"}, {"id": 21, "Type": 2, "Content": "newX(i)=newX(i)-D"}, {"id": 22, "Type": 2, "Content": "i=i+1"}, {"id": 23, "Type": 2, "Content": "newX(i)=newX(i)+D/A(i,i)"}, {"id": 24, "Type": 2, "Content": "i=i+1"}, {"id": 25, "Type": 2, "Content": "i=1"}, {"id": 26, "Type": 2, "Content": "D=X(i,j)*newX(j)"}, {"id": 27, "Type": 2, "Content": "norm=abs(D)"}, {"id": 28, "Type": 2, "Content": "x(i)=newX(i)/D"}, {"id": 29, "Type": 2, "Content": "i=i+1"}, {"id": 30, "Type": 3, "Content": "i<=n"}, {"id": 31, "Type": 2, "Content": "D=X(i,j)*newX(i)"}, {"id": 32, "Type": 2, "Content": "D=abs(D)"}, {"id": 33, "Type": 2, "Content": "norm=norm+D"}, {"id": 34, "Type": 3, "Content": "norm<epsilon"}, {"id": 35, "Type": 2, "Content": "i=it+1"}, {"id": 36, "Type": 3, "Content": "i<iterations"}, {"id": 37, "Type": 5, "Content": "X[n]"}, {"id": 38, "Type": 1, "Content": "End"}
  ],
  "Edges": [
    {"From": 1, "To": 2, "Type": 2}, {"From": 2, "To": 3, "Type": 2}, {"From": 3, "To": 4, "Type": 2}, {"From": 4, "To": 5, "Type": 2}, {"From": 5, "To": 6, "Type": 2}, {"From": 6, "To": 7, "Type": 2}, {"From": 7, "To": 8, "Type": 2}, {"From": 8, "To": 9, "Type": 2}, {"From": 9, "To": 10, "Type": 1}, {"From": 9, "To": 12, "Type": 0}, {"From": 10, "To": 11, "Type": 2}, {"From": 11, "To": 13, "Type": 2}, {"From": 11, "To": 14, "Type": 2}, {"From": 12, "To": 13, "Type": 2}, {"From": 13, "To": 15, "Type": 2}, {"From": 14, "To": 15, "Type": 2}, {"From": 15, "To": 16, "Type": 2}, {"From": 16, "To": 17, "Type": 2}, {"From": 17, "To": 22, "Type": 0}, {"From": 18, "To": 20, "Type": 0}, {"From": 19, "To": 21, "Type": 2}, {"From": 20, "To": 21, "Type": 2}, {"From": 21, "To": 22, "Type": 2}, {"From": 22, "To": 16, "Type": 2}, {"From": 23, "To": 24, "Type": 2}, {"From": 24, "To": 25, "Type": 2}, {"From": 25, "To": 26, "Type": 2}, {"From": 26, "To": 27, "Type": 2}, {"From": 27, "To": 28, "Type": 2}, {"From": 28, "To": 29, "Type": 2}, {"From": 29, "To": 30, "Type": 2}, {"From": 30, "To": 31, "Type": 1}, {"From": 31, "To": 32, "Type": 2}, {"From": 32, "To": 33, "Type": 2}, {"From": 33, "To": 28, "Type": 2}, {"From": 34, "To": 35, "Type": 0}, {"From": 35, "To": 36, "Type": 1}, {"From": 36, "To": 12, "Type": 1}, {"From": 37, "To": 38, "Type": 2}
  ]
}

```

Рис.1. Блок-схема метода Гаусса—Зейделя в формате JSON

```

X(1)=[{"op": "<", "fO": {"op": "+", "fO": {"op": "abs", "od": {"op": "-", "fO": "X0(1)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)", "sO": "X0(2)"}, "sO": "A(1,1)"}}}, "sO": {"op": "abs", "od": {"op": "-", "fO": "X0(2)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(2)"}, "sO": {"op": "*", "fO": "A(2,1)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}}}, "sO": {"op": "abs", "od": {"op": "-", "fO": "B(2)"}, "sO": {"op": "*", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(2,1)"}}}, "sO": "A(2,2)"}}}, "sO": "e"}, {"op": "/", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}, {"op": "-", "fO": {"op": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}, {"op": "-", "fO": {"op": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}, {"op": "<", "fO": {"op": "+", "fO": {"op": "abs", "od": {"op": "-", "fO": "X0(1)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}}}, "sO": {"op": "abs", "od": {"op": "-", "fO": "X0(2)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(2)"}, "sO": {"op": "*", "fO": "A(2,1)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}}}, "sO": {"op": "abs", "od": {"op": "-", "fO": "B(2)"}, "sO": {"op": "*", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(2,1)"}}}, "sO": "A(2,2)"}}}, "sO": "e"}, {"op": "/", "fO": {"op": "-", "fO": "B(2)"}, "sO": {"op": "*", "fO": "A(2,1)"}, "sO": {"op": "/", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}}}, "sO": {"op": "abs", "od": {"op": "-", "fO": "B(2)"}, "sO": {"op": "*", "fO": {"op": "-", "fO": "B(1)"}, "sO": {"op": "*", "fO": "A(1,2)"}, "sO": "X0(2)"}, "sO": "A(1,1)"}}}, "sO": "A(2,2)"}

```

Рис.2. Представление метода Гаусса—Зейделя в форме Q -детерминанта в формате JSON (матрица порядка 2, одна итерация)

ЗАДАЧА 1. Программная Q-система для исследования ресурса параллелизма численных алгоритмов

- ▶ Вторая подсистема для вычисления ресурса параллелизма численных алгоритмов включает базу данных, серверное и клиентское приложения.
- ▶ Для разработки базы данных использовалась СУБД *PostgreSQL*.
- ▶ База данных содержит сущности *Algorithms* и *Determinants*.
- ▶ Атрибуты сущности *Algorithms*: первичный ключ, название алгоритма, описание алгоритма, количество загруженных в базу данных *Q*-детерминантов для различных значений параметров размерности и количества итераций.
- ▶ Атрибуты сущности *Determinants*: первичный ключ, уникальный идентификатор алгоритма, значения параметров размерности, *Q*-детерминант, значения $D_A(\bar{N})$ и $P_A(\bar{N})$, количество итераций.

ЗАДАЧА 1. Программная Q -система для исследования ресурса параллелизма численных алгоритмов

Серверное приложение реализует следующие методы:

- ▶ запись нового алгоритма,
- ▶ обновление информации об алгоритме,
- ▶ получение списка алгоритмов,
- ▶ загрузка нового Q -детерминанта и рассчитанных характеристик $D_{\mathcal{A}}(\bar{N})$ и $P_{\mathcal{A}}(\bar{N})$,
- ▶ сравнение характеристик $D_{\mathcal{A}}(\bar{N})$ или $P_{\mathcal{A}}(\bar{N})$ двух алгоритмов,
- ▶ получение списка Q -детерминантов,
- ▶ скачивание Q -детерминанта,
- ▶ удаление Q -детерминанта,
- ▶ удаление алгоритма вместе с его Q -детерминантами,
- ▶ интерполяция и экстраполяция функций $D_{\mathcal{A}}(\bar{N})$ и $P_{\mathcal{A}}(\bar{N})$.

COMPARE	ID	Name	Description	Number of Q-determinants	ADD
<input checked="" type="radio"/>	1	Gauss – Jordan method	Solution of systems of linear algebraic equations by the Gauss – Jordan method	4	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	2	Algorithm for solving a quadratic equation	Solving a quadratic equation without using a unary minus operation	1	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	3	Gauss – Seidel method	Solution of systems of linear algebraic equations by the Gauss – Seidel method	13	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	4	Algorithm for computing the scalar product of vectors	Calculation of the scalar product of vectors with sequential addition of products	26	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	5	Algorithm for computing the scalar product of vectors	Calculation of the scalar product of vectors with addition of products according to the doubling scheme	26	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	6	Algorithm for solving a quadratic equation	Solving a quadratic equation using unary minus operation	1	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	10	Jacobi method	Solution of systems of linear algebraic equations by the Jacobi method	28	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	11	The algorithm for finding the maximum element in a sequence of numbers	Search for the maximum element in a sequence of numbers	16	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	12	Matrix multiplication algorithm	Matrix multiplication without using the doubling scheme	31	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="radio"/>	13	Matrix multiplication algorithm	Multiplication of matrices using the doubling scheme	31	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>

Рис.3. Главная страница клиентского приложения Q-системы

Q-system

Algorithms > Q-determinants of algorithm #3 "Gauss – Seidel method"

ID	Dimension parameter values	Number of iterations	Algorithm height	Algorithm width	ADD
1	2	1	10	4	
2	2	2	16	10	
3	2	3	22	16	
4	2	4	28	22	
5	2	5	34	28	
6	2	6	40	34	
7	2	7	46	40	
8	2	8	52	46	
9	2	9	58	52	
10	5	1	28	196	
11	5	2	49	5482	
12	5	3	70	145128	
13	10	1	78	16388	

© 2018-2019

Рис.4. Страница клиентского приложения *Q*-системы со списком *Q*-детерминантов и их характеристик

В монографии

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

на стр. 207 написано:

«Использование внутреннего параллелизма имеет очевидное достоинство, т.к. не нужно тратить дополнительные усилия на изучение вычислительных свойств вновь создаваемых алгоритмов. Недостатки также очевидны из-за необходимости определять и исследовать графы алгоритмов. В тех случаях, когда внутреннего параллелизма недостаточно для эффективного использования конкретного параллельного компьютера, приходится заменять его другим алгоритмом, имеющим лучшие свойства параллелизма. Правда, осуществить этот выбор не так легко, т.к. нужно знать параллельную структуру алгоритмов. А она-то как раз неизвестна. Поэтому понятно, насколько актуальными являются сведения о параллельных свойствах алгоритмов, а также знания, позволяющие эти сведения получать».

Концепция *Q*-детерминанта и основанная на ней *Q*-система решают поставленную проблему.

ЗАДАЧА 2. Проектирование эффективных программ, использующих ресурс параллелизма численных алгоритмов полностью

- ▶ Модель концепции Q -детерминанта позволяет исследовать машинно-независимые свойства численных алгоритмов, однако не учитывает особенности их выполнения на параллельных вычислительных системах (ПВС).
- ▶ В связи с этим модель была расширена путем добавления двух подмоделей, которые представляют собой модели параллельных вычислений *PRAM* и *BSP*, отражающие в абстрактной форме архитектуру ПВС с общей и с распределенной памятью соответственно.

ЗАДАЧА 2. Проектирование эффективных программ, использующих ресурс параллелизма численных алгоритмов полностью

На основе расширенной модели был предложен метод проектирования параллельной программы для выполнения Q -эффективной реализации численного алгоритма.

Метод состоит из трех этапов:

1. Построение Q -детерминанта алгоритма.
2. Описание Q -эффективной реализации алгоритма.
3. Если Q -эффективная реализация выполнима, то по ее описанию разрабатывается параллельная программа.

ЗАДАЧА 2. Проектирование эффективных программ, использующих ресурс параллелизма численных алгоритмов полностью

- ▶ Полученная с помощью метода программа называется *Q-эффективной*, а процесс ее создания *Q-эффективным программированием*.
- ▶ *Q-эффективная* программа полностью использует ресурс параллелизма алгоритма, так как выполняет его *Q-эффективную* реализацию.
- ▶ Таким образом, *Q-эффективная* программа имеет самый высокий параллелизм среди программ, реализующих алгоритм.
- ▶ По этой причине *Q-эффективная* программа использует ресурсы ПВС эффективнее, чем программы, выполняющие другие реализации.

ЗАДАЧА 2. Проектирование эффективных программ, использующих ресурс параллелизма численных алгоритмов полностью

Метод был апробирован на алгоритмах, Q -детерминанты которых содержат Q -термы различного типа, в их числе:

- ▶ алгоритм умножения плотных матриц (безусловные Q -термы),
- ▶ алгоритм умножения разреженных матриц (условные Q -термы),
- ▶ метод Гаусса–Жордана для решения СЛАУ (условные Q -термы),
- ▶ метод Якоби для решения СЛАУ (условные бесконечные Q -термы),
- ▶ метод Гаусса–Зейделя для решения СЛАУ (условные бесконечные Q -термы).

ЗАДАЧА 2. Проектирование эффективных программ, использующих ресурс параллелизма численных алгоритмов полностью

- ▶ Для перечисленных алгоритмов и других студентами ЮУрГУ разработаны Q-эффективные программы для общей и распределенной памяти.
- ▶ Использовался язык программирования C++.
- ▶ Для общей памяти применялась технология OpenMP, для распределенной MPI и OpenMP.
- ▶ Исследование проводилось на суперкомпьютере «Торнадо ЮУрГУ».
- ▶ Программы для общей памяти выполнялись на одном процессорном узле, для распределенной памяти на нескольких процессорных узлах.

ЗАДАЧА 3. Повышение эффективности реализации численных методов и алгоритмических проблем

Основные положения технологии повышения эффективности реализации численных методов и алгоритмических проблем состоят в следующем.

- 1) Описание алгоритмической проблемы.**
- 2) Определение методов решения алгоритмической проблемы.**
- 3) Определение классов алгоритмов для реализации каждого из методов и выбор для каждого класса с помощью Q -системы алгоритма с минимальной высотой.**
- 4) Выбор алгоритма с минимальной высотой для реализации алгоритмической проблемы с помощью Q -системы из множества алгоритмов, выбранных из классов алгоритмов, реализующих методы.**
- 5) Разработка Q -эффективной программы для выбранных алгоритмов для реализации методов или алгоритмической проблемы.**

Описанная технология названа Q -эффективным программированием в широком смысле.

ЗАКЛЮЧЕНИЕ

- 1) Приведенные результаты в совокупности являются одним из решений проблемы повышения эффективности параллельных вычислений, использующих численные алгоритмы.
- 2) Их могут применять разработчики программного обеспечения для проектирования эффективных программ, предназначенных для любых ПВС — от персональных компьютеров с многоядерными процессорами до суперкомпьютеров.
- 3) Это приведет к уменьшению времени выполнения программного обеспечения и повышению быстродействия вычислительных систем.
- 4) Новым направлением исследований на основе концепции Q -детерминанта в настоящее время является решение проблемы создания единой для всех численных алгоритмов программной системы для проектирования и исполнения Q -эффективных программ.
Первые результаты данного направления исследований уже получены.

Спасибо за внимание!