УДК 004.932.4

ЭФФЕКТИВНОСТЬ И МАСШТАБИРУЕМОСТЬ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ ПРИ УМНОЖЕНИИ МАТРИЦ

Лямина О. В., Назарова И. А., Фельдман Л.П.
Донецкий национальный технический университет

Рассматривается эффективность и масштабируемость параллельных алгоритмов блочного умножения матриц семейства Кэннона. Строится функция изоэффективности, сравнивается эффективность алгоритмов, анализируются результаты по построенным графикам.

Введение

В данной работе рассматривается семейство алгоритмов Кэннона, которое основано на блочном разбиении матриц.

В алгоритме Кэннона две исходные матрицы A и B и матрица результат C разделяются на блоки. Семейства Кэннона изменяет отображения блоков двух из трех матриц, которые берут участие в вычислении произведения.

Пусть количество столбцов/строк матрицы n кратно числу узлов решётки p. Количество узлов решётки по вертикали/горизонтали равно q. Если представить матрицы в виде квадратных блоков размером $k=n/q$ элементов, то каждому узлу можно однозначно поставить в соответствие такой блок.

1 Алгоритм вычисления матричного произведения с сохранением отображения блоков матрицы-результата *C* (алгоритм №1)

Алгоритм включает в себя шаги:

1. Блоки строк матрицы A сдвигаются циклично влево на i узлов по горизонтали, где i – индекс строки матрицы A.
2. Блоки столбцов матрицы B сдвигаются циклично вверх на j узлов по вертикали, где j – индекс столбца матрицы A.

Алгоритм выполняется за q шагов, где q – размерность вычислительной решётки. Каждый шаг состоит из следующих действий:

1. На вычислительном узле решётки с индексами (i, j) производится умножение блоков A_{ij} и B_{ij}.
2. Циклическое смещение блоков матрицы A влево на 1 узел по горизонтали решётки.
3. Циклическое смещение блоков матрицы B вверх на 1 узел по вертикали решётки.

Результат умножения матриц хранится в матрицы C, блоки которой не подлежат смещению.

2 Алгоритм вычисления матричного произведения с сохранением отображения блоков матрицы A (алгоритм №2)

Алгоритм включает в себя шаги:

1. Блоки столбцов матрицы B сдвигаются циклично вверх на j узлов по вертикали, где j – индекс столбца матрицы B.
2. Блоки строк матрицы B сдвигаются циклично вправо на i узлов по горизонтали, где i – индекс строки матрицы B.
3. Блоки строк матрицы C сдвигаются циклично вправо на i узлов по горизонтали, где i – индекс строки матрицы C.

Алгоритм выполняется за q шагов, где q – размерность вычислительной решётки. Каждый шаг состоит из следующих действий:

1. на вычислительном узле решётки с индексами (*i, j*) производится умножение блоков A_{ij} и B_{ij}.
2. Циклическое смещение блоков матрицы *B* вправо на 1 узел по горизонтали решётки.
3. Циклическое смещение блоков матрицы *B* вверх на 1 узел по вертикали решётки.

Результат умножения матриц хранится в матрицы *C*, блоки которой подлежат смещению. Поэтому по завершению нужно выровнять матрицу до выходного отображения блоков.

3 Алгоритм вычисления матричного произведения с сохранением отображения блоков матрицы B (алгоритм №3)

Алгоритм включает в себя шаги:

1. Блоки строк матрицы *A* сдвигаются циклично влево на *i* узлов по горизонтали, где *i* – индекс строки матрицы *A*.
2. Блоки столбцов матрицы *A* сдвигаются циклично вниз на *j* узлов по вертикали, где *j* – индекс столбца матрицы *A*.
3. Блоки столбцов матрицы *C* сдвигаются циклично вниз на *i* узлов по горизонтали, где *i* – индекс столбца матрицы *C*.

Алгоритм выполняется за *q* шагов, где *q* – размерность вычислительной решётки. Каждый шаг состоит из следующих действий:

1. На вычислительном узле решётки с индексами (*i, j*) производится умножение блоков A_{ij} и B_{ij}.
2. Циклическое смещение блоков матрицы *A* влево на 1 узел по горизонтали решётки.
3. Циклическое смещение блоков матрицы *C* вниз на 1 узел по вертикали решётки.

Результат умножения матриц хранится в матрицы *C*, блоки которой подлежат смещению. Поэтому по завершению нужно выровнять матрицу до выходного отображения блоков.

4 Изоэффективный анализ

Существует несколько подходов для численной оценки свойств масштабируемости параллельного алгоритма в совокупности с архитектурой, на которой он реализован. Наиболее применяемым является метод, указанный в [1-2] и основанный на введении функции изоэффективности. Для этого определяется новая динамическая характеристика: T_O – накладные расходы на параллелизм (*total overhead*). Величина общих накладных расходов включает суммарные затраты всех процессоров параллельной системы, в том числе на последовательную часть распараллеленного алгоритма, реализацию обменов, непродуктивные затраты на синхронизацию и время простоя из-за несбалансированности загрузки процессоров. Для данной параллельной системы T_O рассчитывается по формуле

$$T_O(m,p)=p*T_p-T_1 \quad (1)$$

где T_1 – время, необходимое для выполнения задачи заданного размера на одном процессоре с помощью наилучшего последовательного алгоритма; T_p – общее время выполнения параллельного алгоритма на параллельной архитектуре:

$$T_p = T_{p,comp} + T_{p,comm} \quad (2)$$

Используя введенные обозначения, можно получить новые выражения для времени параллельного решения задачи, ускорения и эффективности:

$$S_p = \frac{T_1}{T_p} = \frac{pT_1}{T_1 + T_o},\ E_p = \frac{S_p}{p} = \frac{T_1}{T_1 + T_o} = \frac{1}{1 + T_o / T_1}. \quad (3)$$

Пусть $E_p = const$ задает уровень эффективности выполнения вычислений, тогда:

$$T_O / T_1 = (1 - E) / E,$$

$$T_1 = E / (1 - E)T_0 = K(p*T_p - T_1) \quad (4)$$

где $K = E / (1 - E)$ – коэффициент, который зависит только от значения показателя эффективности.

Рассчитаем эти показатели для алгоритмов Кэннона.

Для алгоритма вычисления матричного произведения с сохранением отображения блоков матрицы-результата C накладные расходы считаются так:

$$T_o = p(2(t_s + t_w \cdot \frac{n^2}{p}) + \frac{n^3}{p} + 2t_s\sqrt{p} + 2t_w \cdot \frac{n^2}{\sqrt{p}}) - n^3 = 2t_s p(1+\sqrt{p}) + 2t_w n^2(1+\sqrt{p}), \quad (5)$$

$$T_1 = K \cdot T_o = K(2t_s p(1+\sqrt{p}) + 2t_w n^2(1+\sqrt{p})). \quad (6)$$

Если игнорировать второе слагаемое, то получим

$$n^3 = 2t_s p(1+\sqrt{p}). \quad (7)$$

Теперь рассмотрим только второе слагаемое.

$$\begin{aligned} n^3 &= 2t_w n^2(1+\sqrt{p}) \\ n &= 2t_w(1+\sqrt{p}) \\ n^3 &= (2t_w(1+\sqrt{p}))^3 \end{aligned} \quad (8)$$

Можно сказать, что это алгоритм с функцией изоэффективности порядка $O(p^{1,5})$.

Для алгоритма вычисления матричного произведения с сохранением отображения блоков матрицы-результата A и B накладные расходы считаются так:

$$T_o = p(3(t_s + t_w \cdot \frac{n^2}{p}) + \frac{n^3}{p} + 2t_s\sqrt{p} + 2t_w \cdot \frac{n^2}{\sqrt{p}}) - n^3 = t_s p(3+2\sqrt{p}) + t_w n^2(3+2\sqrt{p}), \quad (9)$$

$$T_1 = K \cdot T_o = K(t_s p(3+2\sqrt{p}) + t_w n^2(3+2\sqrt{p})) \quad (10)$$

Зная эти соотношения можно построить графики функции изоэффективности.

Для $t_s = 10$ и $t_w = 3$ для поддержания эффективности $E = 0{,}3$ получаем значения на рис. 1. Черным обозначен алгоритм вычисления матричного произведения с сохранением отображения блоков матрицы-результата *C*, серым – вычисления матричного произведения с сохранением отображения блоков матрицы-результата *A* и *B*. Графики на рисунках 1–3 демонстрируют, какой размерности задачу необходимо решать на имеющейся параллельной вычислительной системе, чтобы эффективность использования оборудования достигала определённого заданного значения.

Для поддержания эффективности $E = 0{,}5$ получаем значения на рис. 2.

Для поддержания эффективности $E = 0{,}7$ получаем значения на рис. 3.

Как видно из графиков при одинаковых значениях эффективности алгоритм вычисления матричного произведения с сохранением отображения блоков матрицы-результата *C* показал лучшее результаты. Это значит, что для достижения одинаковой эффективности для алгоритма №1 нужен меньший размер задачи, чем для алгоритмов №2 и 3 при одинаковом числе используемых процессоров.

Выводы

Применение функции изоэффективности помогает оценить качество полученного параллельного алгоритма, используя одно выражение и связывая размерность задачи и количество процессоров в вычислительной системе. Так на примере было рассмотрено три алгоритма семейства Кэннона. Лучшие результаты показал алгоритм с сохранением отображения блоков матрицы-результата *C*.

Все три алгоритма были реализованы на языке C++, с использованием библиотеки MPI.

Литература

[1] Гергель В.П. Теория и практика параллельных вычислений. – Москва: Бином. Лаборатория

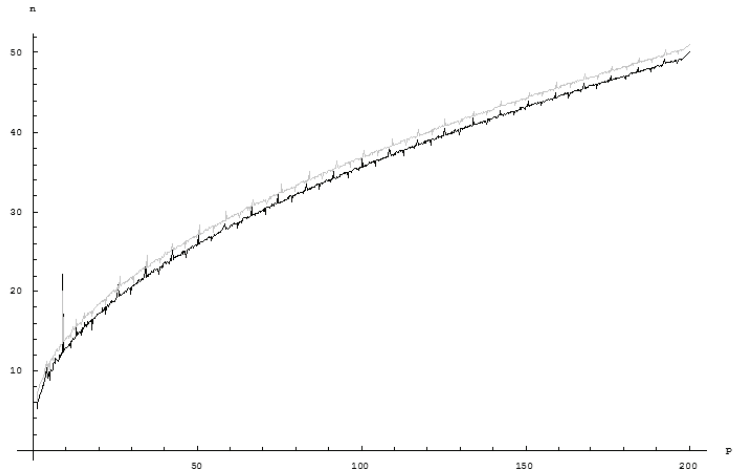

Рисунок 1. График зависимости размера матрицы от количества процессоров, при $E = 0{,}3$

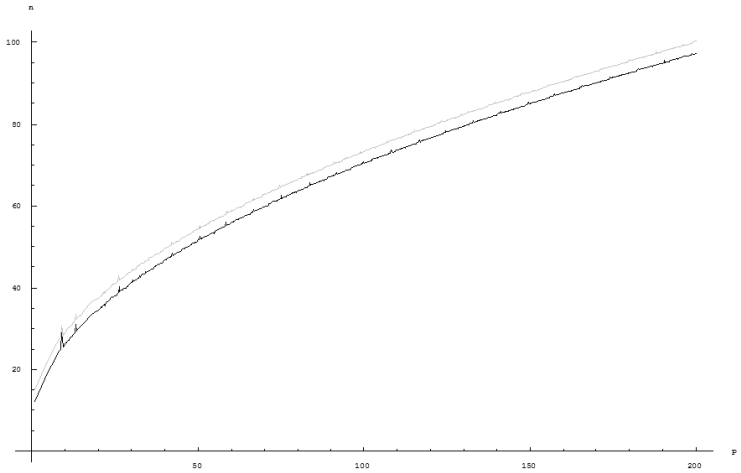

Рисунок 2. График зависимости размера матрицы от количества процессоров, при $E = 0{,}5$

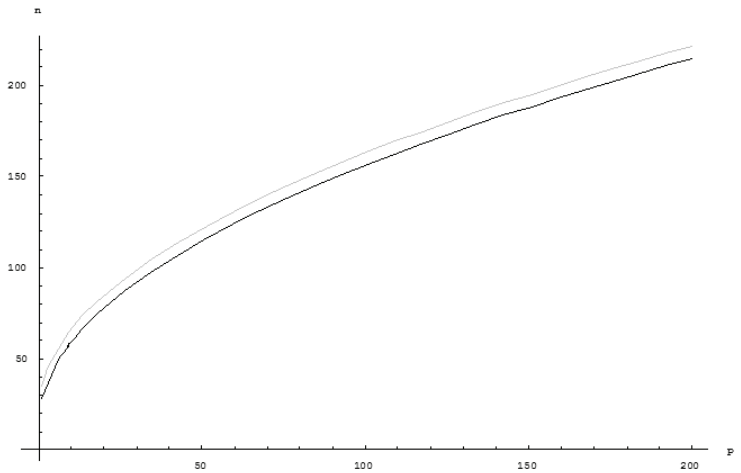

Рисунок 3. График зависимости размера матрицы от количества процессоров, при $E = 0{,}7$

знаний, 2007. – 423с.

[2] Grama A., Gupta A., Kumar V. Isoefficiency: Measuring the scalability of parallel algorithms and architectures // IEEE Parallel and Distributed technology, 2003. – P. 12-21.

[3] Фельдман Л.П., Назарова И.А. Эффективность параллельных алгоритмов оценки локальной апостериорной погрешности для численного решения задачи Коши // Электронное моделирование, т. 29, № 3, 2007. – С. 11-25.