Распараллеливание схемы покомпонентного расщепления для численного решения уравнения теплопроводности

С.А. Ильин, А.В. Старченко

Национальный исследовательский Томский государственный университет

Рассматриваются параллельные алгоритмы численного решения многомерного уравнения теплопроводности с использованием схемы покомпонентного расщепления для вычислительных систем с общей и распределенной памятью. Для распределенных систем реализованы алгоритмы, выполняющие в MPI транспонирование матрицы сеточных переменных, а также применяющие методы параметрической или конвейерной прогонки.

Современные математические модели физических процессов используют, как правило, эволюционные уравнения в частных производных. Снабженные дополнительными начальными и граничными условиями они решаются численно с использованием метода сеток. Для получения численного решения с заданной точностью требуется сгущение узлов вычислительной сетки. Однако чем подробнее сетка, тем больше объем вычислений и дольше длится расчет даже на современных персональных компьютерах. Чтобы эффективно использовать многоядерность и многопроцессорность современных суперкомпьютеров, необходимо разрабатывать специальные алгоритмы для такой вычислительной техники [1, 2].

Рассмотрим двумерную задачу нестационарной теплопроводности в бесконечном бруске прямоугольного поперечного сечения. В начальный момент времени температура бруска имела постоянное значение, затем через боковые поверхности за счет радиационного и конвективного потоков при одной теплоизолированной поверхности начинается нагрев бруска. В этом случае математическая модель рассматриваемого процесса имеет вид:

$$\begin{cases} \dfrac{\partial u}{\partial t} = a^2\left(\dfrac{\partial^2 u}{\partial x^2}+\dfrac{\partial^2 u}{\partial y^2}\right), 0 < x < A, 0 < y < B \\ u(x, y, 0) = 300^\circ K \\ \lambda\dfrac{\partial u}{\partial x}\Big|_{x=0} = \sigma\left(u^4\Big|_{x=0} - T^4\right);\ \lambda\dfrac{\partial u}{\partial x}\Big|_{x=A} = 0;\ \lambda\dfrac{\partial u}{\partial y}\Big|_{y=0} = \alpha_1\left(u\Big|_{y=0} - T\right);\ \lambda\dfrac{\partial u}{\partial y}\Big|_{y=B} = \alpha_2\left(T - u\Big|_{y=B}\right) \end{cases}$$

$$A = 0.1\,\text{м}, B = 0.3\,\text{м}, \lambda = 384\,\frac{\text{Вт}}{\text{м}\cdot\text{K}}, \alpha_1 = 100\,\frac{\text{Вт}}{\text{м}^2\cdot\text{K}}, \alpha_2 = 0.1\,\frac{\text{Вт}}{\text{м}^2\cdot\text{K}}, T = 3000^\circ\text{K}.$$

Для численного решения этой задачи строим сетку (Рис.1), вводим неизвестные значения сеточной функции температуры и, пользуясь методом конечных разностей, неявной аппроксимацией и методом покомпонентного расщепления [3], получаем следующую разностную схему:

$$\begin{cases} \dfrac{u_{ij}^{n+\frac{1}{2}} - u_{ij}^{n}}{\tau} = a^2\,\dfrac{u_{i+1,j}^{n+\frac{1}{2}} - 2u_{ij}^{n+\frac{1}{2}} + u_{i-1,j}^{n+\frac{1}{2}}}{h_x^2},\ i=\overline{1,N-1};\ j=\overline{1,N-1} \\ \dfrac{u_{ij}^{n+1} - u_{ij}^{n+\frac{1}{2}}}{\tau} = a^2\,\dfrac{u_{i,j+1}^{n+1} - 2u_{ij}^{n+1} + u_{i,j-1}^{n+1}}{h_y^2},\ i=\overline{1,N-1};\ j=\overline{1,N-1} \end{cases}$$

с граничными условиями

$$\begin{cases} u_{0,j}^{n+1/2} = \dfrac{\lambda}{\lambda + 4\sigma h_x (u_{0j}^n)^3} u_{1,j}^{n+1/2} + \dfrac{\sigma h_x T^4 + 3\lambda (u_{0j}^n)^4}{\lambda + 4\sigma h_x (u_{0j}^n)^3}; \; u_{N,j}^{n+1/2} = u_{N-1,j}^{n+1/2} \\ u_{i,0}^{n+1} = \dfrac{\lambda}{\lambda + \alpha_1 h_y} u_{i,1}^{n+1} + \dfrac{T\alpha_1 h_y}{\lambda + \alpha_1 h_y}; \; u_{i,N}^{n+1} = \dfrac{\lambda}{\lambda + \alpha_2 h_y} u_{i,N-1}^{n+1} + \dfrac{T\alpha_2 h_y}{\lambda + \alpha_2 h_y}. \end{cases}$$

Цель работы — построение параллельных алгоритмов для решения эволюционных уравнений в частных производных с помощью схемы покоординатного расщепления. Рассматриваются четыре подхода: для многопроцессорных систем с распределенной памятью - алгоритм параметрической прогонки Яненко [4], алгоритм конвейерной прогонки, алгоритм транспонирования, для многоядерных систем с общей памятью - распараллеливание метода прогонки с помощью технологии OpenMP [2].

Заметим, что в параллельных вычислениях много внимания уделялось параллельному решению линейных систем, в частности трехдиагональных систем [1]. В настоящее время наметился переход от мелкозернистых к крупнозернистым алгоритмам для ленточных систем линейных уравнений с узкой лентой. В крупнозернистых алгоритмах выделяют большие подзадач, которые решаются параллельно, в то время как в мелкозернистых параллельно решаются небольшие подзадачи. Например, в статье [5] рассматривается параллельная реализация метода расщепления для уравнения теплопроводности на кластерных системах. Приведены графики зависимости времени работы от числа процессорных элементов. В работе [6] для подобных задач рассматриваются и обосновываются алгоритмы распараллеливания матричной прогонки.

В работе используются две парадигмы параллельного программирования: параллельное программирование с помощью технологии OpenMP и параллельное программирование с помощью стандарта передачи сообщений MPI. Для методов, реализуемых с помощью технологии MPI, проведен теоретический анализ вычислительной сложности алгоритмов и выполнена их практическая реализация.

При конструировании параллельных алгоритмов для вычислительных систем с распределенной памятью используется одномерная декомпозиция (Рис. 1.), при которой сеточная область разделяется на части вдоль сеточных линий j=const.

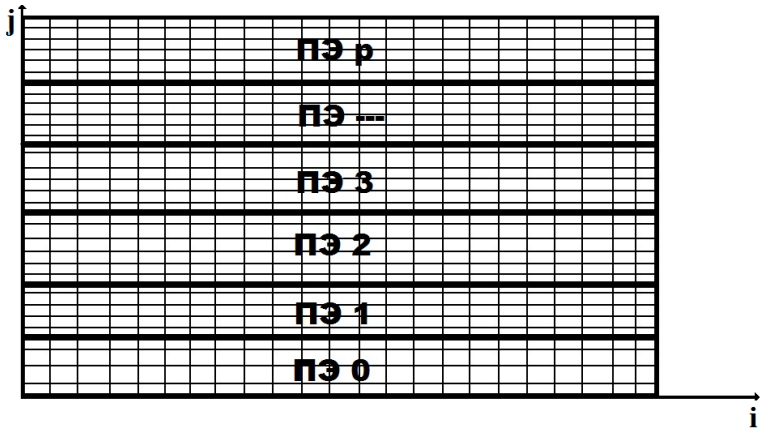

Рис.1. Одномерная декомпозиция

Поэтому на первом дробном шаге, где прогонки выполняются вдоль сеточных линий j=const, пересылок данных между процессорными элементами при реализации метода прогонки не требуется, а на втором дробном шаге, где необходимо трехточечные уравнения решать по сеточным линиям i=const, производится применение алгоритмов, рассмотренных ниже.

Алгоритм параметрической прогонки Н.Н.Яненко [4] осуществляется на основе принципа декомпозиции системы сеточных уравнений, связывающих неизвестные значения сеточной функции вдоль одной сеточной линии i=const. Выделяются так называемые «параметрические» неизвестные, связанные с общими узлами сетки на границе сеточных подобластей $ПЭ_{k-1}$ и $ПЭ_k$. Для этих параметрических неизвестных составляются трехточечные уравнения, которые реша-

ются обычным методом прогонки на каждом процессорном элементе. Затем с помощью явной формулы восстанавливаются значения решения в промежуточных узлах сетки, принадлежащих отдельному процессорному элементу, причем вычисления проводятся одновременно.

В алгоритме транспонирования на втором дробном шаге направления, вдоль которых используется метод прогонки, как бы меняются местами. Т.е. с помощью массовой пересылки между процессорными элементами данных $\{u_{ij}^{n+1/2}\}$ и их транспонирования появляется возможность воспользоваться преимуществами первого дробного шага, когда при реализации на каждом процессорном элементе метода прогонки вдоль сеточных линий j=const не требуется передача данных между процессорными элементами. В данной работе рассматривались два способа совместной передачи и транспонирования двумерного массива с помощью средств стандарта Message Passing Interface. Первый способ опирается на использование функций MPI_ISend и MPI_IRecv. Процедуры MPI_ISend и MPI_IRecv - это процедуры двухточечного неблокирующего обмена, позволяющие совмещать вычисления и передачу сообщений. Второй способ реализует рассматриваемую операцию с помощью процедуры MPI_AlltoAll. Эта процедура позволяет каждому процессорному элементу собирать предназначенные для модификации данные $\{u_{ij}^{n+1/2}\}$ с остальных процессорных элементов и передавать всю полученную информацию специальным образом («транспонируя») каждому из процессорных элементов. На рис.2 представлены графики ускорения и эффективности рассмотренных способов транспонирования матриц 10000x10000 при их одномерной декомпозиции по процессорным элементам. Из рисунка видно, что оба способа транспонирования показывают неплохое ускорение и эффективность, но все-таки второй способ оказывается более выигрышным.

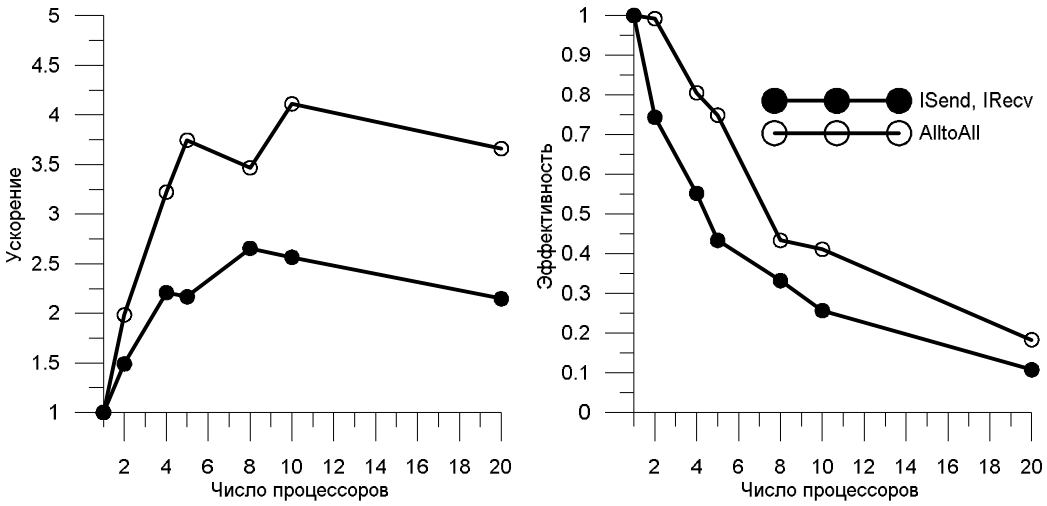

Рис.2. Графики ускорения и эффективности для двух способов транспонирования матрицы 10000x10000

При реализации конвейерной прогонки на втором дробном шаге вдоль сеточных линий i=const (Рис.1) сначала выполняется прямой ход метода прогонки, при котором производится расчет прогоночных коэффициентов последовательно с увеличением индекса j и передачи последних рассчитанных на $ПЭ_k$ прогоночных коэффициентов для начала расчета прогоночных коэффициентов вдоль той же самой сеточной линии на следующем процессорном элементе $ПК_{k+1}$. Обратный ход метода прогонки осуществляется в обратном направлении с передачей последних насчитанных на $ПЭ_k$ значений решения на процессорный элемент $ПЭ_{k-1}$. Поскольку описанная выше процедура является полностью последовательной, для повышения эффективности параллельной MPI-программы в алгоритм была введена конвейеризация: все количество сеточных линий i=const было разделено на блоки по q линий и каждый такой блок обрабатывался (прямой и обратный ход прогонки) на одном процессорном элементе один за другим. Такая организация параллельных вычислений позволяет совмещать вычисления с пересылками данных и повышает эффективность распараллеливания.

Также в работе рассматривался еще один подход, в котором распараллеливание схемы покомпонентного расщепления для многоядерной системы с общей памятью проводится с помо-

щью технологии OpenMP. Это выполняется за счет добавления в программу OpenMP-директив по распараллеливанию циклов, в которых методом прогонки решаются сеточные уравнения.

Расчеты проводились на вычислительном кластере ТГУ СКИФ Cyberia, размер вычислительной задачи составил 4000x4000 узлов. При компиляции параллельных программ использовался компилятор Intel, который позволяет достичь высокой производительности Windows и Unix приложений.

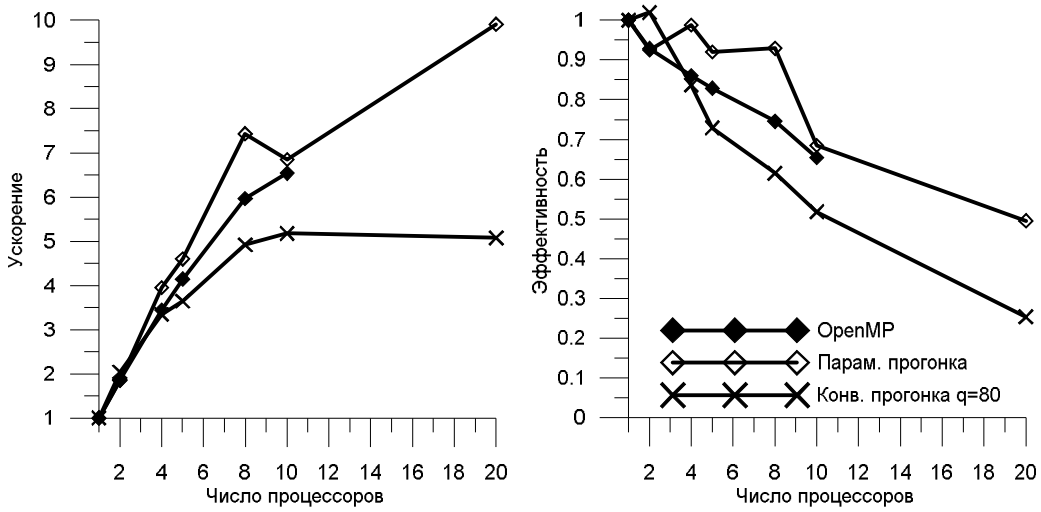

Рис. 3. Ускорение и эффективность различных способов распараллеливания

На рис.3 представлено ускорение и эффективность параллельных программ для разного количества процессоров. Рассматривая представленные графики рис.3, видно, что максимальное ускорение OpenMP-программы примерно равно 6,5. Это довольно хорошее ускорение, если считать, что запуск производился всего на 10 ядрах одного вычислительного узла кластера ТГУ СКИФ Cyberia. Эффективность параллельной OpenMP-программы для этого случая составляет 65%. Более высокие показатели ускорения и эффективности дает применение метода параметрической прогонки при распараллеливании вычислений на втором дробном шаге. На двадцати процессорах эффективность более 50%. Использование конвейерной прогонки на втором дробном шаге дает меньший прирост ускорения параллельной программы даже при выборе оптимального значения параметра q.

Работа выполнена по Государственному Заданию Министерства образования и науки РФ, №5.628.2014/К.

Литература

1. Старченко А.В., Берцун В.Н. Методы параллельных вычислений – Томск: Изд-во Том. ун-та, 2013. – 223с.

2. Высокопроизводительные вычисления на кластерах.- Томск: Изд-во Том. ун-та, 2008. - 200с.

3. Яненко Н.Н. Метод дробных шагов решения многомерных задач математической физике. Новосибирск: Наука, 1967. 197с

4. Яненко Н.Н., Коновалов А.Н., Бугров А.Н., Шустов Г.В. Об организации параллельных вычислений и распараллеливание прогонки // Численные методы механики сплошных сред - 1978 , №7, с. 136-139.

5. Маркус Е.Д. Исследование параллельной реализации метода расщепления для уравнения теплопроводности на кластерных вычислительных системах// Вестник Том. гос. ун-та - 2011, №1.

6. Акимова Е.Н. Распараллеливание алгоритма матричной прогонки// Математическое моделирование - 1994, №9, том 6.