ТРУДЫ
Института механики
им. Р.Р. Мавлютова
Уфимского научного центра РАН

ISSN 2542–0380
Том 11 (2016), № 2, с. 247–253
УДК 532.546; 533.15

http://proc.uimech.org/uim2016.2.036
DOI: 10.21662/uim2016.2.036

Получена: 13.12.2016
Принята: 29.12.2016

Высокопроизводительный алгоритм для решения явных конечно-разностных задач по технологии MPI

Михайленко К.И.

Институт механики им. Р.Р. Мавлютова УНЦ РАН, Уфа

Представлен высокопроизводительный алгоритм для вычислительных систем кластерной архитектуры при использовании технологии обмена сообщениями (MPI). В основу алгоритма положена пространственная декомпозиция области. Показано, что при многоэтапном вычислении каждой пространственной подобласти отдельным процессом можно добиться кратного числу вычислительных процессов линейного роста производительности многопроцессорного приложения с точностью до времени старта/завершения вычислительного процесса. Проведена оценка минимального количества узлов вычислительной сетки, приходящейся на каждый вычислительный процесс, при котором достигается линейный рост производительности.

Ключевые слова: параллельный алгоритм, кластерная вычислительная система, пространственная декомпозиция, математическое моделирование, гидродинамика

1. Введение

Вычислительное моделирование динамики жидкости или газа и, в особенности, многофазных систем, сопряжено со значительными требованиями к вычислительным ресурсам. Особенно ярко увеличение необходимого процессорного времени и потребляемой при моделировании оперативной памяти проявляется при полномасштабном прямом трёхмерном моделировании сложных технических систем. Связано это с тем, что рост количества узловых точек вычислительной сетки пропорционален N^3, что следует из трёхмерности моделируемой области. Также увеличивает требуемые вычислительные ресурсы усложнение современных математических моделей, в которые добавляются сложные многоэтапные физические и химические процессы. Примеры таких «тяжёлых» задач можно видеть в работах [1–3].

Можно выделить два, по бóльшей части пересекающихся, основных подхода к организации высокопроизводительных вычислений. Первый подход связан исключительно с алгоритмическим ускорением вычислительного процесса. Он диктуется задачами, решение которых сопряжено с вычислением систем линейных алгебраических уравнений, и призван снизить вычислительную сложность решения задачи с N^2 до, в идеале, N, как это можно видеть на примере работ [4, 5]. Второй подход связан с экстенсивным ускорением вычислительного процесса при увеличении количества задействованных вычислительных ресурсов. Для задач, вычислительная сложность которых изначально равна N, к которым относятся в том числе и прямые методы решения явный конечно-разностных схем. Для этого подхода существует своё ограничение максимальной производительности связанное с законом Амдаля [6]. Поэтому даже для задач с вычислительной сложностью N необходимо разрабатывать алгоритмы, позволяющие сократить объём последовательной составляющей вычислительного процесса. Примерами работ, связанных с разработкой указанных алгоритмов, являются [7,8], а также данная статья.

В представленной работе описывается класс алгоритмов, связанных с прямым решением явных конечно-разностных численных схем, ориентированных на использование в рамках вычислительных систем кластерного типа. То есть таких, в которых отдельные «независимые» вычислительные блоки связаны между собой относительно медленной средой передачи данных, к тому же, отличаю-

щейся не слишком высокой пропускной способностью по сравнению с шиной данных вычислителя. Следует отметить, что под такое описание в той или иной мере подпадает значительное количество современных суперкомпьютерных систем.

2. Пространственная декомпозиция расчётной области

При распараллеливании явных конечно-разностных схем для вычислений по методу прямого счёта наибольшей популярностью пользуется так называемая пространственная декомпозиция расчётной области (domain decomposition method). Особенно удобна пространственная декомпозиция при использовании кластерных вычислительных систем.

Рассмотрим пример пространственной декомпозиции. Хотя наибольшая выгода распараллеливания проявляется при расчётах трёхмерных областей, все примеры будут приведены для двумерной области по причине большей наглядности представленных в статье схем. Обычно переход в трёхмерную область несложен, сопряжён с добавлением ещё одного индекса в выражениях и не представляет заметных технических проблем.

На рис. 1 показан пример размещения узловых точек при использовании пятиточечной явной вычислительной схемы в двумерной области. Здесь синим обозначены опорные точки на текущем временно́м слое, значения в которых известны, а красным — точка, значение в которой вычисляется. Таким образом, вычисление каждого нового значения в момент времени t^{n+1} по причине явности схемы может рассматриваться как независимая задача вида

$$u_{i,j}^{n+1} = f(u_{i,j}^{n}, u_{i+1,j}^{n}, u_{i,j+1}^{n}, u_{i-1,j}^{n+1}, u_{i,j-1}^{n+1}, h_x, h_y)\,,$$

где u — значения рассчитываемой физической величины в соответствующих узловых точках; h_x и h_y — расстояния между узловыми точками разностной сетки вдоль соответствующих осей; f — некоторая функция, точный вид которой определяется как исходным дифференциальным уравнением для физической величины u, так и методами построения конечно-разностного аналога.

При рассматриваемом подходе к организации вычислений, когда каждое значение на новом шаге по времени может вычисляться независимо, для распараллеливания удобно использовать технологию пространственной декомпозиции области. Идея такой декомпозиции достаточно наглядна и схематически показана на рис. 2.

Рассмотрим данную схему подробнее. Здесь прямоугольником $ABCD$ обозначена вся расчётная

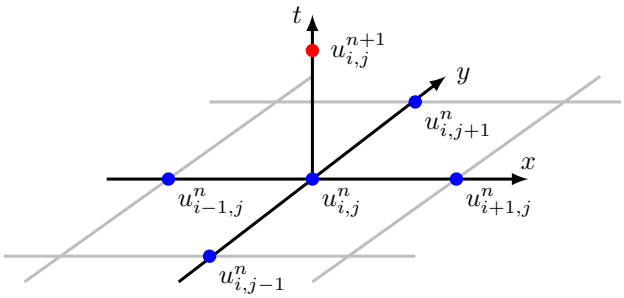

Рис. 1. Схема расположения узловых точек, используемых для прямого расчета нового значения по явной схеме, в двумерном случае

область, в которой и происходит последовательное по временны́м шагам вычисление искомой физической величины u. Серыми линиями E_1E_2, F_1F_2 и G_1G_2 показано разбиение расчётной области вдоль оси x на четыре подобласти AE_1E_2D, $E_1F_1F_2E_2$, $F_1G_1G_2F_1$ и G_1BCG_2, предназначенные для расчётов на отдельных узлах вычислительного кластера каждая. Для простоты буде считать, что на каждом вычислительном узле выполняется только один процесс: p_0, p_1, p_2 и p_3 соответственно, как это обозначено на схеме.

При таком разбиении процесс вычислений на каждом шаге по времени может быть охарактеризован следующими этапами:

- каждый процесс вычисляет новые значения физической величины u на очередном шаге по времени $n+1$, соответствующие выделенной ему пространственной подобласти;

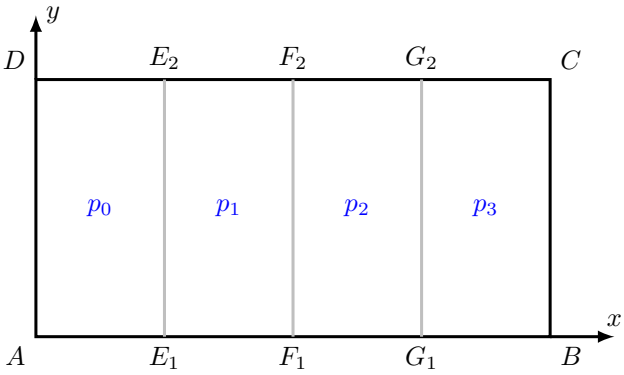

Рис. 2. Схема линейной пространственной декомпозиции двумерной расчётной области вдоль оси x на четыре подобласти, предназначенные для расчёта на процессах $p_0 \ldots p_3$ соответственно

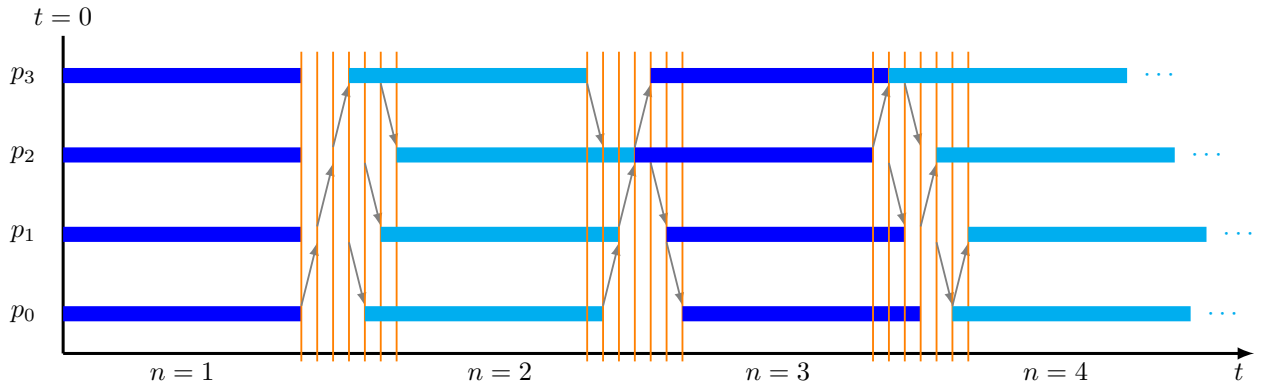

Рис. 3. Пример временно́й диаграммы параллельного вычислительного процесса при линейной пространственной декомпозиции расчётной области для работы на четырёх узлах кластерной системы

- каждая пара соседних процессов обменивается актуальной информацией о вычисленных приграничных значениях (так называемые теневые грани), что необходимо будет для перехода к следующему шагу по времени;
- объявляем вновь вычисленные значения значениями на текущем шаге по времени $(n+1 \to n)$ и повторяем описанный вычислительный процесс до достижения заданного времени или условия.

Узким местом описанного вычислительного процесса является момент обмена теневыми гранями. В это время вычисления остановлены и процессы ждут информации от соседей о значениях величины u в приграничной области; вычисления могут продолжить выполнение лишь после получения указанной информации. Таким образом, с точки зрения параллельного алгоритма, процесс передачи теневых граней должен рассматриваться как последовательная часть вычислительного кода, что, в соответствии с законом Амдаля, ограничивает максимально достижимое ускорение вычислений.

Указанное ограничение наиболее заметно проявляется при использовании небольших и недорогих локальных кластеров, на которых в качестве среды передачи данных используется локальная сеть Ethernet. Отличительными особенностями Ethernet, отрицательно влияющими на скорость параллельных вычислений, являются значительное время установления соединения между парой узлов (latency time $t_\ell \leqslant 150$ мкс) и ограничение на количество одновременно устанавливаемых соединений, число которых может варьироваться от единицы до $p-1$, где p — число процессов параллельного приложения.

Если предположить наихудший вариант, когда одновременно возможна передача только одного сообщения, то диаграмма загруженности узлов вычислительного кластера получится приблизительно такой, как это показано на рис. 3. Здесь показаны четыре шага по времени. Время, в течение которого процессы заняты непосредственно вычислением, отмечено чередующимися синими и голубыми линиями. Оранжевые вертикальные лини отсекают моменты начала и окончания передачи теневых граней между процессами, а сами передачи и их направления показаны стрелками.

Из представленной диаграммы видно, что при любых вариантах передачи данных время T_s (время последовательных вычислений), когда процессы заняты не вычислениями, а передачей данных друг другу, пропорционально времени единичной передачи t_s и числу процессов: $T_s \sim pt_s$. Таким образом, если обозначить через T_p время вычисления отдельным процессом своей подобласти, то максимально достижимое ускорение рассматриваемого параллельного процесса будет ограничено величиной $(T_p+T_s)/T_s$ вне зависимости от количества вычислительных устройств.

Для наглядности: если параллельное приложение использует 32 процесса, вычисление одной подобласти занимает 10 с, а передача теневой грани с учётом времени установления соединения занимает около 200 мкс, то легко видеть, что невозможно ускорить вычисления более, чем в 2,5 раза.

3. Алгоритм с расщеплением вычисления подобластей

В значительной мере избежать описанных выше ограничений возможно, если организовать вычислительный процесс таким образом, чтобы при вычислении процессам не было необходимости ожидать получения данных. То есть, реорганизовать последовательность счёта таким образом, чтобы данные от процесса к процессу передавались до пе-

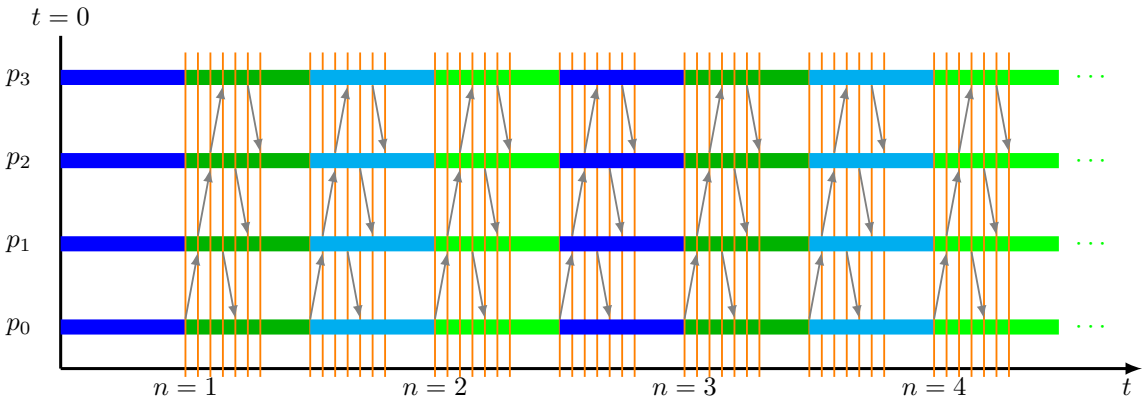

Рис. 4. Временна́я диаграмма параллельного вычислительного процесса при линейной пространственной декомпозиции расчётной области с расщеплением подобластей для работы на четырёх узлах кластерной системы

рехода к следующему шагу вычислений.

На рис. 4 представлена диаграмма работы алгоритма с расщеплением подобластей, как это изображено на рис. 5. Здесь показан пример расщепления четырёх подобластей, ранее представленных на рис. 2. На представленной схеме дополнительно проведена оранжевая линия, делящая каждую подобласть на две части. Если теперь, после вычисления каждой части подобласти, организовать пересылку соответствующей вычисленной части теневой грани, то возможно таким образом подобрать размеры подобластей, чтобы свести к минимуму время ожидания получения данных ($T_s \to 0$).

Последовательность работы по алгоритму с расщеплением подобластей показана на рис. 4, где проиллюстрированы четыре шага по времени, причём время расчёта каждым процессом своей подобласти разделено на две части. Синей и зелёной

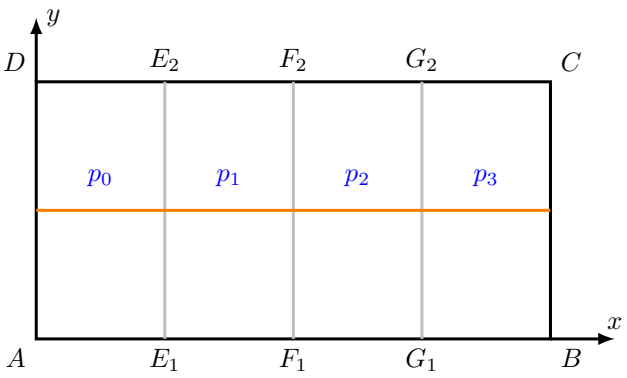

Рис. 5. Схема линейной пространственной декомпозиции двумерной расчётной области вдоль оси x на четыре подобласти с расщеплением

линиями показаны времена расчёта первой и второй частей соответственно (а для чётных шагов по времени — светло-синей и светло-зелёной). Предполагается, что после расчёта каждой половины подобласти процессы вызывают функции неблокирующей передачи для пересылки уже рассчитанной половины теневой грани. Время начала и конца передач обозначено оранжевыми вертикальными линиями, а направление передач — стрелками.

Представленный на рис. 4 процесс может быть описан в виде следующей последовательности шагов:

- процессы начинают вычисления соответствующего шага по времени одновременно;
- после расчёта половины подобласти (расчёт до оранжевой линии на рис. 5) каждый из процессов вызывает неблокирующие передачи для пересылки соседям уже рассчитанных половин теневых граней;
- так как передача осуществляется неблокирующими функциями, процессы без задержки продолжают вычисление второй половины подобласти;
- неблокирующая передача будет осуществлена системой при доступности среды передачи данных;
- если время расчёта половины подобласти t_p превышает время передачи всех сообщений ($t_p \geqslant 2(p-1)t_s$), то переход к вычислению следующего шага по времени будет произведён без задержек.

4. Оценка эффективности алгоритма с расщеплением

Исходя из вышесказанного, для задачи с заданным количеством узловых точек, и учитывая технические данные вычислительной системы, такие, как производительность процессора, скорость передачи данных и время установления отдельного соединения, возможно оценить оптимальное количество процессов p, при котором полученное ускорение будет близко к указанному p.

Часто для обеспечения непрерывности вычислений отдельный процесс выделяют для организации ввода-вывода. Тогда формально условие сохранения линейного ускорения, кратного числу процессов, может быть записано в виде неравенства, связывающего время расчета одной части подобласти и время передачи одного сообщения:

$$t_p \geqslant 2(p-1)t_s + pT_{\text{io}}\,,$$

здесь T_{io} — время передачи результатов расчёта одной подобласти выделенному процессу для сохранения.

При выполнении записанного выше условия все обмены данными между узлами вычислительной системы будут разнесены во времени и не будут интерферировать со временем непосредственного счёта.

Оценим входящие в представленное условие параметры.

Время вычисления одной части подобласти может быть оценено по формуле

$$t_p = \frac{IJm}{\nu}\,.$$

Здесь I и J — размеры части подобласти в точках (узлах вычислительной сетки); m — количество операций, требующихся для расчета всех параметров в одной точке на одном временном слое (для случая системы уравнений Навье–Стокса в двумерной области эта величина может быть оценена в $m \approx 10^2$); ν — производительность процессора в Mflops.

Время пересылки оценивается как

$$t_s = t_\ell + \frac{Jk}{u}\,,$$

где t_ℓ — время латентности среды передачи данных; k — размер числа в байтах; u — скорость передачи данных по сети.

Добавка, учитывающая время периодической передачи результатов расчета нулевому процессу для сохранения на диске

$$T_{\text{io}} = \frac{1}{\mu}\left(\frac{2IJk}{u} + t_\ell\right)\,.$$

Здесь μ — частота пересылки промежуточных результатов.

Пусть $I = J$ для упрощения оценки. В этом случае, на основе приведенных уравнений, записывается квадратное уравнение относительно I, которое имеет положительный вещественный корень только при выполнении условия

$$\left[(p-1)\frac{k}{u}\right]^2 \geqslant \left(\frac{pk}{\mu u} - \frac{m}{\nu}\right)\left[2(p-1)t_\ell + \frac{pt_\ell}{2\mu}\right]\,.$$

Последнее условие можно записать в виде оценки:

$$\frac{pk}{\mu u} \leqslant \frac{m}{\nu}\,.$$

5. Заключение

Представленный в работе алгоритм позволяет значительно снизить последовательную составляющую параллельного приложения, которое реализует прямой метод решения задач динамики сплошной среды и математической физики, использующих явный конечно-разностный метод дискретизации исходных дифференциальных уравнений.

При использовании итогового соотношения можно получить оценку допустимого размера подобласти при пространственной декомпозиции и, с учётом числа процессов, — всей расчётной области. При этом получаемые величины доказываются в разумных и удобных для организации вычислительного процесса пределах ($I \sim 100 \div 1000$).

Список литературы

[1] Михайленко К.И., Везиров Р.Р., Ахатов И.Ш., Урманчеев С.Ф. Численное моделирование течения мелкодисперсного катализатора в канале лифт-реактора // Нефтепереработка и нефтехимия. 1997. № 12. С. 17–20.

[2] Михайленко К.И., Валеева Ю.Р. Моделирование осаждения мелкодисперсной взвеси из воздуха при прохождении волн давления // Вычислительные методы и программирование: новые вычислительные технологии. 2013. Т. 14, № 1. С. 328–333.

[3] Михайленко К.И., Кулешов В.С. Математическое моделирование скоростной неравномерности потока газа за пористой преградой // Вычислительные технологии. 2015. Т. 20, № 6. С. 46–58.

[4] Марьин Д.Ф., Малышев В.Л., Моисеева Е.Ф., Гумеров Н.А., Ахатов И.Ш., Михайленко К.И. Ускорение молекулярно-динамических расчетов с помощью быстрого метода мультиполей и графических процессоров // Вычислительные методы и программирование: новые вычислительные технологии. 2013. Т. 14, № 1. С. 483–495.

[5] Moiseeva E, Mikhaylenko C, Malyshev V, Maryin D, Gumerov N. FMM/GPU Accelerated Molecular Dynamics Simulation of Phase Transitions in Water-Nitrogen-Metal Systems. ASME. ASME International Mechanical Engineering Congress and Exposition, Volume 7: Fluids and Heat Transfer, Parts A, B, C, and D. Pp. 883–892.

[6] Gene M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities // Proceedings of the April 18-20, 1967, spring joint computer conference (AFIPS '67 (Spring)). Pp. 483–485.

[7] Газизов Р.К., Лукащук С.Ю., Михайленко К.И. Разработка параллельных алгоритмов решения задач механики сплошной среды на основе принципа пространственной декомпозиции // Вестник Уфимского государственного авиационного технического университета. 2003. Т. 4, № 1. С. 100–107.

[8] Gazizov R.K., Khizbullina S.F., Lukashuk S.Yu., Mikhaylenko C.I. Numerical Solving of Fluid Dynamics Equations on Cluster Computing Systems: a Technique Using Domain Decomposition // Proc. of the 4th International Workshop on Computer Science and Information Technologies, CSIT'2002. Patras, Greece. 2002. Pp. 281-286.

The high-performance technique for solving an explicit finite-difference problems on the MPI technology

Mikhaylenko C.I.

Mavlyutov Institute of Mechanics, Ufa

The technique for high-performance computing cluster architecture using messaging passing interface (MPI) is presented. The technique is based on the spatial domain decomposition. It is shown that when calculating multistage each spatial subregion separate process can achieve a multiple number of computational processes linear growth performance multiprocessor applications up to the start time / end of the computational process. An assessment of the minimum amount of computational grid points attributable to each computational process, at which the linear increase in performance.

Keywords: parallel technique, MPI, domain decomposition method, numerical simulation, CFD