Разработка параллельного алгоритма LU-SGS для решения многомерных задач вычислительной газодинамики

И. В. Семенов, И. Ф. Ахмедьянов
Институт автоматизации проектирования РАН, Москва

Введение

Современные многомерные задачи вычислительной газовой динамики требуют значительного количества вычислительных ресурсов. Решение таких задач за приемлемое время невозможно без использования высокопроизводительных параллельных компьютеров. Для целого ряда задач, например стационарных, применение неявных численных схем даёт возможность существенно сократить время расчета. Но переход к неявной схеме вычислений ведет к существенному усложнению построения алгоритма распараллеливания. Прежде всего, это связано с тем, что в результате линеаризации возникает система линейных уравнений большого порядка, параллельное решение которой сопряжено с рядом трудностей.

Представленная работа посвящена разработке параллельного алгоритма Lower Upper — Symmetric Gauss-Seidel для сильносвязанных кластеров с распределенной памятью. Последовательный вариант алгоритма описан в работе [1]. В работе [2] в качестве параллельного алгоритма LU-SGS предлагается гибридный алгоритм, который соответствует последовательному алгоритму только в определенных подобластях исходной расчетной области. В данной работе при распараллеливании осуществляется строгое соблюдение последовательного алгоритма LU-SGS.

Последовательный алгоритм

Будем называть ячейкой сетки выпуклый многогранник и значения искомых функций в области, ограниченной этим многогранником. Естественным образом с каждой ячейкой сопоставляется список ее непосредственных геометрических соседей. Солвер — функция, в качестве параметров принимающая некоторую текущую ячейку с ее геометрическими соседями,

и модифицирующая значения искомых функций в текущей ячейке и соседних. Обход — процесс последовательного применения солвера к ячейкам сетки. Под обходом так же будем понимать некоторую последовательность ячеек сетки.

Таким образом, если рассмотреть алгоритм LU-SGS как объект для распараллеливания, то его последовательный вариант выглядит как два прохода (прямой и обратный) по ячейкам расчетной области. Оба прохода осуществляются по одной последовательности ячеек, но в разных направлениях.

В общем случае значения искомых функций в ячейке на следующем временном слое зависят от всех ячеек расчетной области на текущем временном слое, что затрудняет построение параллельного алгоритма.

Параллельный алгоритм

Построим параллельный алгоритм LU-SGS для сильносвязанных кластеров с распределенной памятью.

Так как обходить солвером расчетную область можно в любом порядке (не обязательно, чтобы следующая ячейка в обходе была геометрическим соседом предыдущей), то построение параллельного алгоритма LU-SGS сводится к выбору последовательного обхода области, который можно корректно декомпозировать на параллельные обходы. То есть обходы, выполняющиеся одновременно на разных процессорах кластера в распределенной памяти. Результат параллельного выполнения таких обходов идентичен результату последовательного.

В данной работе сначала строятся параллельные обходы подобластей, после чего показывается возможность сведения их в один обход. Структура параллельных обходов эффективно отображается на распределенную память.

Разбиение области (поддомены, внутренние/внешние ячейки)

Все множество ячеек области разбивается на N подмножеств, где N — количество участвующих в расчете процессоров кластера. Каждое такое подмножество назовем поддоменом.

Внутри каждого из построенных поддоменов ячейки сетки разбиваются на два класса. Внутренние ячейки — ячейки, не име-

ющие общих граней с ячейками других поддоменов, и внешние ячейки — ячейки, имеющие хотя бы одну общую грань с ячейкой из другого поддомена. Ячейка, не имеющая общую грань с ячейкой другого поддомена, но имеющая грань-границу расчетной области считается внутренней.

Изолированность обходов

Два обхода называются изолированными, если множества их ячеек являются подмножествами разных поддоменов расчетной области, и результат их параллельного расчета в распределенной памяти идентичен результату последовательного расчета в единой памяти.

Пусть K_1 и K_2 — два множества ячеек. $N(K) = K \cup N(K)$, где $N(K)$ — множество, состоящее из всех геометрических соседей ячеек из K. Пусть $N(K_1) \cap N(K_2) = \varnothing$.

Так как во время расчета ячеек из K_1 изменению могут подвергнуться только параметры в ячейках из $N(K_1)$, аналогично при расчете K_2 измениться могут только параметры в ячейках из $N(K_2)$, и $N(K_1) \cap N(K_2) = \varnothing$, то два обхода K_1 и K_2 являются изолированными.

M обходов называются изолированными, если они являются попарно изолированными.

Построение разбиения множества всех поддоменов

Необходимо построить разбиение множества всех поддоменов Q на Q_i такое, что внешние ячейки поддоменов из множества Q_i образуют параллельные изолированные обходы (это означает, что ячейки поддоменов из Q_i можно считать параллельно).

Назовем два поддомена i и j зависимыми, если существуют различные ячейки a_1, a_2 и a_3 такие, что:

a_1 – геометрический сосед a_2, a_2 — геометрический сосед a_3;

a_1 принадлежит поддомену i, a_3 принадлежит поддомену j.

Из определения следует, что любые два параллельных обхода внутри независимых поддоменов являются изолированными.

Построим функцию $G(M)$. Пусть M, D — некоторые множества поддоменов области, $R(M,D)$ – функция, возвращающая поддомен с наименьшим порядковым номером из множества M,

такой, что он независим от любого поддомена из множества D (порядок нумерации поддоменов может быть любой).

Пусть M_0 некоторое непустое множество поддоменов, построим множества M_i следующим образом:

$$M_1 = \{R(M_0, \varnothing)\}, M_2 = M_1 \cup \left\{R\left(M_0 \Big/ M_1, M_1\right)\right\}, ...,$$

$$..., M_q = M_{q-1} \cup \left\{R\left(M_0 \Big/ M_{q-1}, M_{q-1}\right)\right\}$$

Построение множеств останавливается, когда функция R не может найти поддомен, удовлетворяющий ее условиям.

Тем самым построена функция $G(M_0) = M_q$, которая выбирает из множества M_0 некоторое достаточно большое подмножество попарно независимых поддоменов M_q. Из непустого множества функция G выбирает хотя бы один узел.

Алгоритм построения Q_i — следующий:

$$Q_1 = G(Q);$$

$$Q_2 = G\left(Q \Big/ Q_1\right);$$

...

$$Q_n = G\left(Q \Big/ (Q_1 \cup ... \cup Q_{n-1})\right);$$

$$Q_{n+1} = G\left(Q \Big/ (Q_1 \cup ... \cup Q_n)\right) = G(\varnothing) = \varnothing$$

Построение множеств останавливается, когда $Q_{n+1} = \varnothing$, что эквивалентно $Q_1 \cup ... \cup Q_n = Q$. Такой момент всегда наступает, так как множество поддоменов Q конечно, а каждая итерация добавляет в множество $Q_1 \cup ... \cup Q_i$ хотя бы один поддомен. Таким образом построено искомое разбиение, удовлетворяющие следующим условиям:

$$\cup Q_i = Q$$

$$Q_i \cap Q_j = \varnothing, i \neq j$$

Внешние ячейки поддоменов из множества Q_i образуют параллельные изолированные обходы. Порядок обхода внешних ячеек внутри одного поддомена не важен.

Идея поэтапного вычисления

Базовой идеей описываемого алгоритма является двухэтапный расчет, когда на первом этапе производится расчет внутренних ячеек области, на втором внешних. Таким образом, для расчета первого этапа не требуются каких-либо пересылок ячеек между узлами кластера. На первом этапе расчета организуется N параллельных обходов расчетной области, с тем условием что эти обходы идут только по внутренним ячейкам поддоменов (порядок внутри каждого поддомена не важен). По определению эти N обходов являются изолированными.

Здесь можно предложить простейший вариант параллельного алгоритма, когда для расчета второго этапа вся расчетная область собирается на одном процессоре. На этом процессоре последовательно рассчитываются внешние ячейки всех поддоменов, после чего измененные ячейки рассылаются на соответствующие узлы. Однако в этом случае, если внешних ячеек всего 5 % от общего количества ячеек, то время работы параллельной реализации составит не менее 5 % от времени работы последовательной. Что соответствует максимальному ускорению не более чем в 20 раз (закон Амдала).

Расчет второго этапа

Хотя на первом этапе производится параллельный расчет внутренних ячеек, последовательный расчет второго этапа сильно снижает эффективность параллельного алгоритма. Поэтому расчет второго этапа производится итерациями $P_1, ..., P_n$. Во время итерации P_i параллельно рассчитываются внешние ячейки поддоменов из множества Q_i. Это означает, что процессоры соответствующие поддоменам из множества Q_i на i-ой итерации обходят солвером свои внешние ячейки. Процессоры соответствующие поддоменам из $Q \setminus Q_i$ простаивают.

Сведение построенных обходов в один

Так как итерация P_i — это расчет некоторого количества изолированных обходов, то ее можно подменить последовательным обходом внешних ячеек поддоменов из множества Q_i. Расчет первого этапа алгоритма также эквивалентен некоторому после-

довательному обходу множества всех внутренних ячеек. Объединение $n+1$ последовательных обходов в один тривиально. Таким образом, найдена параллельная декомпозиция некоторого последовательного обхода области.

Обратный обход

Обратный обход строится следующим образом. Рассчитываются итерации $P_1, ..., P_n$, после чего рассчитывается первый этап алгоритма с обходом ячеек в обратном порядке. Внутри каждого параллельного обхода ячейки обрабатываются в обратном порядке. Нетрудно заметить, что сведение этих параллельных обходов в один даст корректный обратный обход области. Следовательно, предложенный алгоритм позволяет параллельно считать обратный обход.

Пересылки между процессорами кластера

Как уже сказано выше, пересылки необходимы только во время расчета второго этапа алгоритма. Так как итерации P_i выполняются последовательно, то достаточно привести стратегию пересылок ячеек между процессорами кластера внутри одной итерации. Для итерации P_i назовем процессоры, соответствующие множеству Q_i, ведущими, а соответствующие множеству $Q \backslash Q_i$, — ведомыми.

С точки зрения обменов расчет итерации P_i разбивается на 3 стадии:

- ведущие процессоры получают от ведомых необходимые ячейки;
- ведущие процессоры обходят свои внешние ячейки солвером, ведомые процессоры простаивают;
- ведущие процессоры передают обратно измененные ячейки.

Необходимые ячейки — это внешние ячейки ведомых процессоров, являющиеся геометрическими соседями ячеек, которые планируется рассчитать во время итерации P_i.

Из построения множеств Q_i следует, что в пределах итерации P_i ни одна ячейка ведомых процессоров не может быть передана более чем одному ведущему процессору.

Идеальное теоретическое ускорение

Рассмотрим какое максимальное ускорение возможно при использовании данного алгоритма на идеальном кластере, где время передачи данных равно нулю.

Пусть расчетная область равномерно разбита на N подобластей и в каждой подобласти доля внешних ячеек к внутренним равна α.

Тогда для расчета первого этапа алгоритма требуется

$$\frac{(1-\alpha)t}{N}$$

t — время расчета всех ячеек в последовательном алгоритме, $\alpha t/N$ — время расчета внешних ячеек в одном поддомене. Пусть n — это количество итераций второго этапа алгоритма. Так как внутри итерации поддомены из множества Q_i рассчитываются параллельно, то время расчета n итераций составляет

$$\frac{\alpha t}{N} n$$

что равно времени расчета второго этапа. Стоит отметить, что это время не зависит от количества процессоров, участвующих в расчете итерации.

Таким образом теоретическое ускорение можно оценить как

$$\frac{N}{1+\alpha(n-1)}$$

Оценим значение параметра α для трехмерной области при $1 \le N \le M/8$. Пусть область равномерно разбита на кубические поддомены и количество ячеек в области равно M. В каждом поддомене одинаковое количество ячеек. Тогда длину ребра поддомена можно оценить как $\sqrt[3]{M/N}$, в поддоменах без граничных ячеек внешних ячеек будет $6\left(\sqrt[3]{M/N}-1\right)^2$, количество внешних ячеек области $6\left(\sqrt[3]{M}-1\right)^2+2$ и

$$\alpha = \frac{6N\left(\sqrt[3]{M/N}-2\right)^2 + 2N - 6\left(\sqrt[3]{M}-2\right)^2}{M}$$

Предположив, что допустимой считается утилизация ресурсов кластера не ниже 50 % и $n=7$ — характерное значение для трехмерных задач, получаем нижнюю оценку для количества ячеек на один процессор кластера в 45 000 ячеек.

Нетрудно заметить, что в асимптотике, когда $\alpha \to 1$, идеальное ускорение стремится к N/n.

Заключение

Реализация предложенного алгоритма хорошо себя зарекомендовала в проводимых расчетах [3]. Возможно улучшение алгоритма, если строить множества Q_i не на основе внешних ячеек, а на основе более мелких групп ячеек, что позволит повысить параллелизм.

Работа выполнена при поддержке РФФИ грант № 05-08-50115-а и гранта Президента РФ № МК-5068.2007.1

Литература

1. **Menshov, Nakamura Y.** Hybrid Explicit-Implicit, Unconditionally Stable Scheme for Unsteady Compressible Flows // AIAA Journal. — 2004. — Vol. 42. — № 3. — P. 551—559.

2. **Sharov D., Luo H., Baum J. D., Loehner R.** Implementation of unstructured grid GMRES-SGS method on shared-memory, cache-based parallel computers // AIAA-2000-927 — Aerospace Sciences Meeting and Exhibit, 38th, Reno, NV, Jan. — 2000. — P. 10—13.

3. **Семенов И. В., Ахмедьянов И. Ф., Уткин П. С.** Разработка вычислительного комплекса для решения двух и трехмерных задач газодинамики реагирующих течений на многопроцессорных ЭВМ // Высокопроизводительные параллельные вычисления на кластерных системах : Мат-лы 6 Международ. науч.-практ. семинара. 12–17 декабря 2006 г. — СПб. : Изд-во Санкт-Петербургского госуниверситета, 2007. — Т. 2. — С. 138—145.