УДК 519.6

ДВУХУРОВНЕВОЕ РАСПАРАЛЛЕЛИВАНИЕ В МОДЕЛИ СМЕШАННОЙ ПАМЯТИ ДЛЯ РАСЧЕТА ЗАДАЧ ГАЗОДИНАМИКИ В МЕТОДИКЕ ТИМ-2D

А. А. Воропинов, С. С. Соколов, И. Г. Новиков
(РФЯЦ-ВНИИЭФ)

Представлено описание схемы распараллеливания счета задач газовой динамики по методике ТИМ-2D. Рассмотрены алгоритмы распараллеливания по областям в модели распределенной памяти с использованием интерфейса MPI и распараллеливания в модели общей памяти с использованием интерфейса OpenMP. При расчете одной задачи методы распараллеливания могут использоваться как совместно, так и раздельно. Приводятся замеры эффективности на одной из задач.

Введение

Методика ТИМ-2D [1] предназначена для расчета нестационарных двумерных задач механики сплошной среды на нерегулярных лагранжевых сетках произвольной структуры. По этой методике можно проводить расчеты на сетках, состоящих из несамопересекающихся многоугольных ячеек; в узлах сетки может сходиться произвольное количество ребер.

Для расчета задач газодинамики и упругопластичности используются явные конечно-разностные схемы. Кинематические величины относятся к узлам счетной сетки, термодинамические величины — к центрам ячеек.

При решении сложных двумерных задач начальную геометрию системы часто приходится разбивать на несколько счетных областей. Такое разбиение необходимо для более точного описания взаимодействующих тел с выделенными линиями скольжения, называемыми контактными границами (КГ). Методика ТИМ-2D позволяет проводить расчеты в многообластной постановке [2]. При этом предполагается, что разные математические (счетные) области могут взаимодействовать друг с другом вдоль КГ [3]. В сложных задачах могут возникать особые точки, в которых сходятся несколько КГ. Эти точки называются тройными. В начале счетного шага рассчитываются контактные взаимодействия, которые затем, при расчете математических областей, выступают в качестве граничных условий. Расчет самих областей производится независимо друг от друга.

Для повышения точности проводимых расчетов и более адекватного численного моделирования течений и процессов, рассматриваемых в двумерных задачах, часто необходимо выбирать сетки с большим количеством ячеек. В последовательном режиме расчет сложных задач занимает много календарного времени, поэтому более целесообразно проводить его в параллельном режиме. В настоящее время наиболее распространенными стандартами для распараллеливания являются интерфейс передачи сообщений MPI [4] для модели распределенной памяти и интерфейс OpenMP [5], предназначенный для модели общей памяти.

Большинство современных вычислительных систем разрабатываются как кластеры. При этом внутри одного вычислительного узла используется несколько процессоров, функционирующих на общем поле оперативной памяти (так называемая общая память). Использование общей памяти позволяет производить распараллеливание без декомпозиции данных по процессорам (и без формирования параобластей — см. разд. 1), а распараллеливание выполнять внутри счетных циклов, распределяя между процессорами итерации. Такой подход хорош для методик, использующих нерегулярные сетки, так как в них все счетные алгоритмы построены поточечно, без использования упорядоченности структуры сетки. Поэтому для таких ме-

тодик распараллеливание в модели общей памяти в большинстве случаев можно осуществить, не меняя алгоритмов последовательного счета.

Для больших задач ("невмещающихся" в память одного вычислительного узла или требующих больших вычислительных ресурсов) можно использовать смешанную модель распараллеливания (одновременное использование MPI и OpenMP); при этом MPI используется для взаимодействия между вычислительными узлами, а внутри узла распараллеливание осуществляется на общей памяти с использованием интерфейса OpenMP. Это позволяет значительно повысить количество задействованных процессоров при проведении расчетов.

1. Принципы распараллеливания задач газовой динамики и упругопластичности

Для методики ТИМ-2D реализуется трехуровневое распараллеливание в модели смешанной памяти. На верхнем (первом) уровне осуществляется распараллеливание счета по счетным областям (крупноблочное распараллеливание). На среднем (втором) уровне счет распараллеливается по параобластям внутри счетной области (мелкозернистое распараллеливание). На первых двух уровнях используется модель распределенной памяти с использованием интерфейса передачи сообщений MPI. На нижнем (третьем) уровне осуществляется распараллеливание счета внутри области в модели общей памяти с использованием интерфейса OpenMP.

При расчете выделяются головной процессор задачи и головной процессор для каждой области. Головной процессор задачи обеспечивает инициализацию задачи и общее управление счетом. Головной процессор области производит инициализацию области и декомпозицию области для мелкозернистого распараллеливания. При счете этот процессор управляет расчетом области и участвует в расчете контактного взаимодействия вдоль КГ области. Головные процессоры являются логическими, их функции могут быть как распределены по различным физическим процессорам, так и сосредоточены на одном процессоре или иметь какое-то более сложное распределение. Головные процессоры, помимо управления, используются и в вычислениях.

При расчете задач в начале счетного шага рассчитываются КГ и тройные точки. Каждая КГ рассчитывается на одном из двух процессоров, к которым отнесены области, образующие эту границу. Расчет тройной точки дублируется на всех процессорах, рассчитывающих области, окружающие эту тройную точку. Затем КГ выступают в качестве границ при расчете математических областей. Расчет областей выполняется полностью независимо. Это обстоятельство используется для распараллеливания счета по областям. При этом обмены необходимы только на этапе расчета контактного взаимодействия.

При мелкозернистом распараллеливании каждая область разбивается на набор *компактов*. Каждая ячейка относится к одному и только одному компакту. Компакты разных областей не связаны друг с другом. Декомпозиция данных по компактам осуществляется по ячейкам сетки при помощи библиотеки ParMeTiS [6].

На основе каждого компакта формируется *параобласть*. Между соседними параобластями одной математической области используется наложение в один слой ячеек. К одному процессору может быть отнесено несколько параобластей, полученных на основе компактов от разных математических областей. Формат представления параобласти такой же, что и для математической области, поэтому при проведении расчетов на ней используются те же самые счетные алгоритмы с некоторыми дополнениями, связанными с обработкой приграничных и присоединенных точек.

Для решения задачи контактного взаимодействия фрагменты сторон КГ (сторона КГ — это последовательность точек одной области, лежащих на данной КГ) собираются на головных процессорах образующих ее математических областей и расчет КГ производится так же, как при распараллеливании по областям. Данные по рассчитанным границам рассылаются процессорам, рассчитывающим параобласти, примыкающие к КГ.

При использовании интерфейса OpenMP [7] счетные циклы, итерации которых могут быть выполнены независимо друг от друга, распараллеливаются по узлам или ячейкам. Такой подход позволяет использовать интерфейс OpenMP как совместно с распараллеливанием по областям и с мелкозернистым распараллеливанием, так и независимо.

Методы распараллеливания могут использоваться как одновременно, в различных комбинациях друг с другом, так и каждый в отдельности. Выбор режима распараллеливания зависит от решаемой задачи и характеристик используемой ЭВМ.

2. Распараллеливание по счетным областям

На верхнем уровне распараллеливание осуществляется по математическим областям в модели распределенной памяти. Такой метод распараллеливания (без использования мелкозернистого распараллеливания) имеет ряд положительных особенностей:

1. Не требуется дополнительной декомпозиции данных по процессорам, используется существующее разбиение на математические области.
2. Для расчетов используются те же алгоритмы, что и в последовательном режиме счета. Не требуется существенной модификации последовательных программ: счетные программы остаются без изменений, а модификации подвергаются только управляющие программы.

Распараллеливание по счетным областям реализовано, например, в рамках методики ДМК [8].

Количество используемых процессоров и декомпозиция данных для распараллеливания по областям. При распараллеливании по областям (без использования мелкозернистого распараллеливания) максимальное количество вычислительных узлов, которые можно задействовать, равно количеству областей. Разбиение задачи на счетные области выполняется на этапе расчета начальных данных. Из-за различного количества точек, веществ, используемых приближений и счетных процессов в областях объем вычислений, приходящихся на каждую область, может существенно различаться. По этой причине при отнесении к каждому вычислительному узлу по одной области эффективность распараллеливания может оказаться невысокой. Для того чтобы решить эту проблему, в рамках методики ТИМ-2D реализована возможность отнесения нескольких счетных областей к одному MPI-процессу (или вычислительному узлу), т. е. на одном узле может рассчитываться не одна счетная область, а несколько. Благодаря этой возможности, как правило, удается сбалансировать вычислительную нагрузку. При этом уменьшается количество задействованных процессоров, но ускорение остается приблизительно на таком же уровне, что и с использованием максимального количества процессоров.

Рассмотрим вопрос о декомпозиции и количестве используемых процессоров на примере. Пусть область решения задачи состоит из 6 областей. В областях соответственно 100, 50, 40, 30, 10 и 5 тыс. ячеек, общее количество ячеек в задаче — 235 тыс. Для простоты будем считать, что среднее время на расчет одной ячейки во всех областях одинаково. Распараллеливание в модели общей памяти не учитывается. При распараллеливании по областям ускорение не может быть больше, чем в 2,35 раза, так как самая большая область состоит из 100 тыс. ячеек. Используя декомпозицию на два процессора следующего вида: `/1,0,0,0,1,1/` (области 1, 5 и 6 отнесены к процессору 1, области 2—4 — к процессору 0), получаем оценку ускорения сверху 1,96 с эффективностью распараллеливания 98 % (здесь в оценках не учитывается время на организацию параллельных вычислений и обмен данными). При декомпозиции на три процессора вида `/1,0,2,2,0,0/` (область 1 отнесена к процессору 1, области 2, 5, 6 — к процессору 0, области 3, 4 — к процессору 2) получаем оценку ускорения сверху 2,35, а эффективность 78,3 %. При использовании большего количества процессоров оценка ускорения останется той же.

Сложность при выполнении декомпозиции в автоматическом режиме в методике ТИМ-2D связана с тем, что некоторые счетные алгоритмы требуют различного объема вычислений на различных этапах расчета, причем объем вычислений может меняться в широких пределах на протяжении даже нескольких счетных шагов. Одним из таких алгоритмов, например, является коррекция счетной сетки [9]. Для того чтобы избежать влияния конкретного счетного шага, декомпозиция осуществляется на основе информации о времени счета 100 шагов. Исходя из этой информации минимизируется максимальная загрузка процессоров. Такая оценка производится при декомпозиции на двух и более вычислительных узлах (до максимально возможного количества, равного количеству областей). На основе полученных оценок количество процессоров для расчета задачи выбирается так, чтобы получить максимальное ускорение, но при этом выполнять расчет с достаточно высокой эффективностью. В представленном примере максимальное ускорение достигается на трех процессорах, но эффективность оказывается при этом не слишком высокой. Для оптимального использования вычислительной системы можно провести расчет на двух процессорах с небольшим уменьшени-

ем ускорения, но с эффективностью, близкой к 100 %.

Расчет границ между математическими областями. Один из основных вопросов при распараллеливании счета по математическим областям связан с расчетом контактного взаимодействия между областями и расчетом тройных точек (точек, в которых сходится более двух КГ). Так, в методике ДМК [8] расчет контактного взаимодействия осуществляется на головном процессоре задачи. В результате на каждом счетном шаге появляется последовательный участок, головной процессор значительно перегружен обменами, а во время расчета КГ все процессоры, кроме головного процессора задачи, простаивают. Таким образом, расчет КГ становится "узким местом", что приводит к снижению эффективности распараллеливания.

В методике ТИМ-2D это обстоятельство учтено и распараллеливание реализовано также при решении задачи контактного взаимодействия. В ТИМ-2D расчет границы между двумя математическими областями производится на одном из двух процессоров, рассчитывающих области, формирующие эту границу. После этого информация о границе передается второму процессору. За счет того что у каждой области несколько границ (не менее 2, а, как правило, 4 или более), расчет границ распараллеливается из-за распределения областей по разным процессорам.

Расчет тройной точки требует небольшого объема вычислений и поэтому дублируется на всех процессорах, рассчитывающих области, окружающие эту тройную точку.

Расчет временного шага в параллельном режиме. Организация расчета в параллельном режиме имеет несколько особенностей, связанных с необходимостью выполнения обменов, которые описываются ниже. Расчет задачи состоит из последовательного выполнения временных шагов, пока не будет достигнуто условие завершения расчета. Кроме того, в конце любого временного шага могут выполняться различные сервисные операции, такие как печать выдач, сохранение файла-разреза и др.

Непосредственно расчет временного шага состоит из следующих основных этапов:

1. Расчет контактного взаимодействия между математическими областями. Расчет нового положения КГ и тройных точек.
2. Расчет математических областей с использованием информации о положении КГ (КГ выступают в качестве границ математических областей).
3. Завершение текущего счетного шага, формирование данных для следующего счетного шага.
4. Формирование таблицы прерываний и их обработка.

Опишем алгоритмы выполнения этих шагов в режиме с распараллеливанием в модели распределенной памяти.

Расчет контактного взаимодействия. Расчет КГ между двумя математическими областями выполняется на одном из двух процессоров, рассчитывающих области, формирующие эту КГ (обмены производятся только между этой парой процессоров). Если КГ разделяет две области, рассчитываемые на одном и том же процессоре, то КГ рассчитывается на этом процессоре без выполнения обменов, так же, как в последовательном режиме. Внешние границы, которые для простоты также будем называть КГ, рассчитываются на том же процессоре, что и соответствующая им счетная область, без выполнения обменов.

КГ формируется из граничных точек областей, ее образующих. Последовательность точек одной области вдоль КГ называется стороной КГ. Внутренние КГ состоят из двух сторон, внешние — из одной. В параллельном режиме стороны разделяются на *свои* и *чужие* в зависимости от того, как распределены области по процессорам. Сторона КГ является своей для текущего процессора, если образующая ее область рассчитывается на этом процессоре, и чужой в противном случае. В последовательном режиме все стороны свои.

Расчет контактного взаимодействия в параллельном режиме производится по следующему алгоритму (блоки 2, 4, 6, 8 выполняются в параллельном режиме счета):

1. Формирование своих сторон для каждой КГ.
2. Обмен информацией о сторонах КГ (свои отправляются, чужие принимаются).

   Все последующие операции выполняются только для тех КГ, которые рассчитываются на текущем процессоре.
3. Формирование КГ из сторон.
4. Обмен между процессорами информацией о сформированных КГ.

5. Расчет ускорений и скоростей для граничных узлов своих областей.
6. Получение процессором, рассчитывающим КГ, информации о чужой стороне КГ (эта информация подготавливается и отправляется процессором, рассчитывающим соседнюю область).
7. Расчет контактного взаимодействия вдоль КГ. Получение новых координат и скоростей точек КГ.
8. Пересылка рассчитанной КГ.
9. Расчет тройных точек.

Расчет математических областей. После расчета КГ расчет математических областей производится так же, как в последовательном режиме, за исключением того, что счетная область относится строго к одному процессору (рассчитывается только этим процессором). Никаких обменов на этом этапе расчета не производится.

Завершение счетного шага. При завершении счетного шага *заголовочная* информация обо всех областях (*шапка*) передается головному процессору задачи для определения следующего счетного шага, а также формирования таблицы прерываний и ее обработки.

Головной процессор задачи принимает шапки всех областей, формирует данные для следующего счетного шага и таблицу прерываний. Таблица прерываний рассылается всем процессорам. Каждый головной процессор области принимает и обрабатывает таблицу прерываний, и, если она не пустая, выполняется соответствующая сервисная функция, например печать выдачи, сохранение файла-разреза, завершение счета. Если после обработки очередного прерывания были выполнены какие-то сервисные операции (кроме, естественно, завершения счета), то головной процессор задачи снова формирует таблицу прерываний и рассылает ее остальным процессорам. Так продолжается до тех пор, пока не будет разослана пустая таблица прерываний — в этом случае начинается расчет следующего временного шага.

3. Распараллеливание в модели общей памяти с использованием интерфейса OpenMP

Для распараллеливания на нижнем уровне используется модель общей памяти и интерфейс OpenMP. Это распараллеливание базируется на следующих основных принципах:

1. Распараллеливание итераций счетных циклов, таких как циклы по узлам и ячейкам счетной сетки. Такое распараллеливание наиболее просто реализуется в модели общей памяти и позволяет использовать OpenMP совместно с другими способами распараллеливания.
2. Независимое распараллеливание каждого счетного блока. При этом внутри каждого счетного блока количество параллельных областей OpenMP минимизировано. Распараллеливаются счетные блоки, функционирующие в рамках математической области. Фрагменты программы между счетными блоками выполняются последовательно. В последовательном режиме выполняются переход от области к области, расчет контактного взаимодействия, выполнение сервисных операций.

Следование этим принципам позволяет использовать OpenMP как отдельно, так и совместно с распараллеливанием в модели распределенной памяти. При распараллеливании по областям распараллеливаются счетные циклы по ячейкам и узлам области. При мелкозернистом распараллеливании в параллельном режиме выполняются те же самые циклы, но в рамках параобласти (внутреннее представление параобласти такое же, как счетной области).

Для реализации распараллеливания в начале каждого счетного блока и перед счетными циклами, которые должны выполняться параллельно, добавляются директивы OpenMP. Эти директивы являются специальными комментариями, которые игнорируются при проведении расчетов без использования OpenMP.

4. Распараллеливание в модели смешанной памяти

При распараллеливании в модели смешанной памяти в рамках каждого вычислительного узла выполняется один процесс. Каждый такой процесс выступает в качестве MPI-процесса и имеет номер (как это принято в стандарте MPI [4]) от 0 до $N-1$, где N — общее количество используемых вычислительных узлов для расчета задачи. MPI-процесс управляет расчетом задачи в рамках вычислительного узла, производит MPI-об-

мены, является *головной нитью* для OpenMP-распараллеливания [5] в рамках этого узла. MPI-процесс выполняет цикл по счетным областям задачи, но расчет областей, отнесенных к другим узлам, пропускает.

В рамках каждого вычислительного узла MPI-процесс производит порождение параллельных нитей OpenMP для проведения расчетов областей, отнесенных к данному вычислительному узлу (и только этих областей). Областью может являться как математическая область, так и параобласть, в зависимости от режима проведения расчета. Таким образом, OpenMP-распараллеливание никак не зависит от распараллеливания в модели распределенной памяти.

5. Исследование эффективности распараллеливания

Исследование эффективности распараллеливания проводилось по двум основным направлениям:

1) в зависимости от режимов распараллеливания и количества процессоров;
2) в модели общей памяти в зависимости от количества счетных точек.

В качестве характеристик эффективности распараллеливания использовались следующие функции:

$S_p = \dfrac{t_1}{t_p}$ — ускорение счета;

$E_p = \dfrac{t_1}{pt_p} \cdot 100\,\%$ — эффективность распараллеливания,

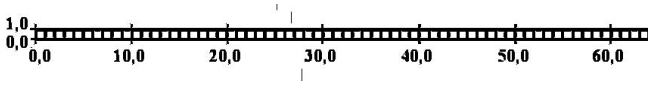
где t_1 — время расчета на одном процессоре используемой параллельной ЭВМ (последовательный режим счета), t_p — время счета на p процессорах.

Исследование эффективности распараллеливания на различном количестве процессоров в различных режимах счета. Для исследования эффективности двухуровневого распараллеливания в различных режимах счета была использована задача об обжатии цилиндра равномерным давлением.

Рассматривается цилиндр радиусом 1 и высотой 64. Все границы, кроме боковой поверхности цилиндра, являются *жесткими стенками*. На боковой поверхности цилиндра задается постоянное по времени давление $P_{\text{гр}} = 5$. Вещество — идеальный газ с показателем адиабаты $\gamma = 3$.

Начальные условия: плотность равна 1, скорость, энергия и давление равны 0.

Область решения разбивалась на 64 счетные области размером 1×1. Разбиение на области приведено на рисунке. Области нумеруются слева направо от 1 до 64. В качестве счетной сетки выбрана четырехугольная счетная сетка. В каждой области бралась равномерная сетка из 10 000 ячеек.



Разбиение на области задачи об обжатии цилиндра

Результаты расчета задачи в разных режимах и на различном количестве процессоров полностью совпадают.

Данные о времени проведения расчетов сгруппированы в таблицы: в табл. 1, 2 — расчеты без использования OpenMP (в модели распределенной памяти с распараллеливанием по областям), в табл. 3 — с использованием OpenMP (расчету в модели общей памяти соответствует строка для 8 процессоров, расчетам в модели смешанной памяти — строки для числа процессоров от 16 до

Таблица 1

Характеристики расчетов в модели распределенной памяти (MPI)

Кол-во процессоров	Общее время счета, с	Ускорение, число раз	Эффективность, %
1	5 397,7	1,00	100
2	2 731,2	1,98	98,813
4	1 427,2	3,78	94,55
8	770,8	7,00	87,54
16	377,9	14,28	89,27
32	201,0	26,80	83,8
64	110,3	48,92	76,44

Таблица 2

Потери эффективности при максимальном количестве обменов

Кол-во процессоров	Время счета, с		Ускорение, число раз		Эф-ность, %		Потери эф-ти, %
	max	min	max	min	max	min	
4	1 518	1 427	3,6	3,8	88,9	94,5	5,6
8	824	771	6,6	7,0	81,9	87,5	5,6
16	386	378	14,0	14,3	87,5	89,3	1,8
32	229	201	23,6	26,8	73,8	83,8	10,0

Таблица 3

Характеристики расчетов в модели смешанной памяти (MPI + OpenMP)

Кол-во процессоров (MPI × OpenMP)		Время расчета, с	Ускорение, число раз	Эффективность, %
1	(1 × 1)	5 397,6	1	100
8	(1 × 8)	1 250,2	4,317	53,97
16	(2 × 8)	615,927	8,763	54,77
32	(4 × 8)	295,725	18,252	57,03
64	(8 × 8)	165,455	32,623	50,97
128	(16 × 8)	97,237	55,510	43,37
256	(32 × 8)	59,605	90,557	35,37
512	(64 × 8)	39,735	135,8	26,53

512). В таблицах приведены значения эффективности распараллеливания, полученные путем замеров времени расчета 2 000 шагов на одном и нескольких процессорах.

Распараллеливание в модели распределенной памяти показывает достаточно высокую эффективность на различном количестве процессоров. Например, на 64 процессорах эффективность составляет 76,5 % (см. табл. 1).

При отнесении двух соседних областей к одному и тому же процессору при вычислении контактного взаимодействия обмен не производится. В табл. 2 приведены замеры времени счета и эффективности распараллеливания при проведении расчетов в модели распределенной памяти в режиме, когда за одним процессором закрепляются соседние и несоседние области. В первом случае расчеты проводятся с минимальным количеством обменов, во втором — с максимальным. При декомпозиции с максимальным количеством обменов (см. табл. 2) при расчете в рамках одного вычислительного узла, состоящего из 8 процессоров, эффективность падает на 5,6 %. На 16 процессорах падение эффективности составляет менее 2 %, а на 32 процессорах достигает 10 %. Что касается расчета на 16 процессорах, то необходимо отметить, что повышение эффективности по сравнению с 8 процессорами происходит в обоих режимах. По-видимому, это связано с влиянием кэш-памяти.

Как видно из результатов, представленных в табл. 3, эффективность распараллеливания в модели общей памяти (для 8 процессоров) оказывается низкой — 54 %. Это объясняется большим количеством КГ в данном расчете (193 КГ, из них 63 — КГ между областями), а расчет КГ в методике ТИМ-2D в модели общей памяти в настоящее время не распараллелен.

Необходимо отметить, что при расчете аналогичной задачи в однообластной постановке (без расчета КГ между математическими областями) в модели общей памяти эффективность составляет около 90 %. Поэтому для повышения эффективности необходимо выполнять распараллеливание расчета КГ в модели общей памяти.

Заметим также, что с увеличением общего количества процессоров до 64 эффективность распараллеливания остается практически постоянной. Это происходит во многом благодаря тому, что расчет КГ в этом случае распараллеливается за счет MPI. С другой стороны, это позволяет надеяться, что распараллеливание с использованием OpenMP не будет ухудшать эффективности распараллеливания, достигнутой при использовании MPI. В целом можно отметить, что в случае распараллеливания расчета КГ можно ожидать эффективности не меньше, чем в режиме счета на распределенной памяти.

Исследование эффективности распараллеливания в модели общей памяти в зависимости от количества счетных точек. Исследование эффективности распараллеливания в модели общей памяти направлено главным образом на выбор оптимального размера компакта для получения максимальной общей эффективности в режиме трехуровневого распараллеливания в модели смешанной памяти.

Для исследования эффективности использовалась тестовая задача о точечном взрыве [10]. Расчеты проводились с различным количеством точек на двух типах сеток (четырехугольная и шестиугольная). Результаты замеров приведены в табл. 4, 5. Жирным шрифтом в таблицах выделены характеристики расчетов, эффективность распараллеливания в которых оказалась выше 95 %.

Как видно из таблиц, для достижения эффективности распараллеливания более 90 % необходимо формировать параобласти с количеством точек от 2 до 10 тыс. вне зависимости от типа используемой сетки.

Заключение

Дано описание метода трехуровневого распараллеливания, разрабатываемого для методики ТИМ-2D. Приводятся первые результаты двухуровневого распараллеливания в модели сме-

Таблица 4

Эффективность распараллеливания в расчетах на шестиугольной сетке

Кол-во ячеек	Время расчета на 1 процессоре, с	Время расчета на 8 процессорах, с	Ускорение	Эффективность, %
100	0,040	0,017	2,353	29,41
500	0,215	0,050	4,300	53,75
1 000	0,522	0,082	6,366	79,57
2 000	1,121	0,155	7,232	90,40
3 000	**2,000**	**0,259**	**7,722**	**96,53**
4 000	**3,096**	**0,365**	**8,482**	**106,03**
5 000	**4,087**	**0,504**	**8,109**	**101,36**
6 000	**5,077**	**0,622**	**8,162**	**102,03**
7 000	**6,432**	**0,817**	**7,873**	**98,41**
8 000	8,444	1,133	7,453	93,16
10 000	10,689	1,451	7,367	92,08
11 000	11,580	1,680	6,893	86,16
12 000	13,277	1,977	6,716	83,95
13 000	15,070	2,310	6,524	81,55
14 000	17,010	2,643	6,436	80,45
15 000	19,242	3,019	6,374	79,67
20 000	26,622	4,300	6,191	77,39
40 000	75,450	12,158	6,206	77,57
80 000	197,048	31,586	6,238	77,98
120 000	132,420	21,330	6,208	77,60
250 000	268,387	42,128	6,371	79,63
500 000	528,460	80,520	6,563	82,04

Таблица 5

Эффективность распараллеливания в расчетах на четырехугольной сетке

Кол-во ячеек	Время расчета на 1 процессоре, с	Время расчета на 8 процессорах, с	Ускорение	Эффективность, %
1 000	0,266	0,058	4,586	57,33
1 500	0,515	0,07	7,357	91,96
2 000	**0,755**	**0,094**	**8,032**	**100,40**
3 000	**0,9**	**0,102**	**8,824**	**110,29**
5 000	**1,615**	**0,187**	**8,636**	**107,95**
8 000	**2,794**	**0,349**	**8,006**	**100,07**
10 000	**3,838**	**0,441**	**8,703**	**108,79**
11 000	**4,042**	**0,511**	**7,910**	**98,87**
12 000	4,316	0,583	7,403	92,54
18 000	6,46	0,985	6,558	81,98
30 000	8,352	1,298	6,435	80,43
40 000	12,092	1,915	6,314	78,93
80 000	34,29	5,37	6,385	79,82
120 000	54,606	8,841	6,176	77,21
250 000	127,763	20,216	6,320	79,00
500 000	289,806	46,121	6,284	78,55

шанной памяти, замеры эффективности в различных режимах распараллеливания на количестве процессоров до 512. Реализованное распараллеливание позволяет рассчитывать задачи более чем в 100 раз быстрее по сравнению с последовательным режимом счета.

В дальнейшем планируется осуществить распараллеливание расчета КГ в модели общей памяти и реализовать алгоритмы мелкозернистого распараллеливания.

Список литературы

1. *Соколов С. С., Воропинов А. А., Новиков И. Г. и др.* Методика ТИМ-2D для расчета задач механики сплошной среды на нерегулярных многоугольных сетках с произвольным количеством связей в узлах // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2006. Вып. 4. С. 29—43.
2. *Воропинов А. А., Соколов С. С., Новиков И. Г.* Расчет контактного взаимодействия между счетными областями в методике ТИМ-2D // Сб. аннотаций докл. VI науч.-тех. конф. "Молодежь в науке". Саров, 30 октября — 01 ноября 2007 г. Саров: РФЯЦ-ВНИИЭФ, 2007. С. 14.
3. *Соколов С. С.* Метод расчета двумерных нестационарных упругопластических течений на нерегулярных многоугольных лагранжевых сетках // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2004. Вып. 4. С. 62—80.
4. MPI Documents. http://www.mpi-forum.org/docs/docs.html 13.09.2007.
5. OpenMP Specifications. http://www.openmp.org/drupal/node/view/8 13.09.2007.
6. ParMETIS — Parallel Graph Partitioning and Fill-reducing Matrix Ordering. http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview 13.09.2007.
7. *Воропинов А. А., Новиков И. Г., Соколов С. С.* Использование интерфейса OpenMP для распараллеливания методики ТИМ // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2007. Вып. 3—4. С. 74—82.

8. *Мотлохов В. Н., Рассказова В. В., Софронов И. Д.* Об опыте распараллеливания счета задач газовой динамики по программам ДМК на мультипроцессорных системах // Там же. 1996. Вып. 3. С. 3—8.

9. *Новиков И. Г., Панов А. И., Соколов С. С.* Способ коррекции нерегулярной лагранжевой сетки методом наложения дифференцируемых связей // Журнал вычисл. мат. и мат. физ. 2005. Т. 45, № 8. С. 1487—1500.

10 *Бондаренко Ю. А., Воронин Б. Л., Делов В. И. и др.* Описание системы тестов для двумерных газодинамических методик и программ. Ч. 1. Требования к тестам. Тесты 1—7 // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 1991. Вып. 2. С. 3—9.

Статья поступила в редакцию 18.10.07.