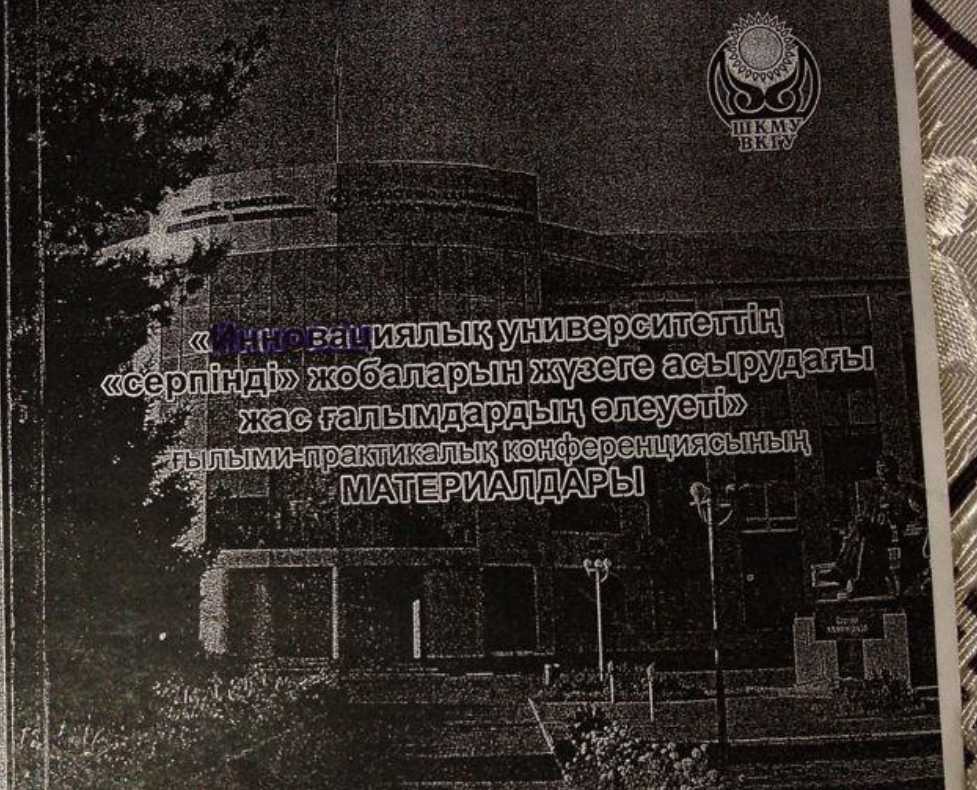
ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
С. АМАНЖОЛОВ АТЫНДАҒЫ ШЫҒЫС ҚАЗАҚСТАН МЕМЛЕКЕТТІК УНИВЕРСИТЕТІ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
ВОСТОЧНО-КАЗАХСТАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМ. С. АМАНЖОЛОВА

ШҚМУ
ВКГУ



«Инновациялық университеттің «серпінді» жобаларын жүзеге асырудағы жас ғалымдардың әлеуеті»

ғылыми-практикалық конференциясының

МАТЕРИАЛДАРЫ

МАТЕРИАЛЫ

научно-практической конференции

«Потенциал молодых ученых в реализации «прорывных» проектов Инновационного университета»

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
С. АМАНЖОЛОВ АТЫНДАҒЫ ШЫҒЫС ҚАЗАҚСТАН МЕМЛЕКЕТТІК УНИВЕРСИТЕТІ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
ВОСТОЧНО-КАЗАХСТАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМ. С. АМАНЖОЛОВА

«Инновациялық университеттің «серпінді» жобаларын жүзеге асырудағы жас ғалымдардың әлеуеті»

ғылыми-практикалық конференциясының

МАТЕРИАЛДАРЫ

I-БӨЛІМ

МАТЕРИАЛЫ

научно-практической конференции

«Потенциал молодых ученых в реализации «прорывных» проектов инновационного университета»

ЧАСТЬ I

Өскемен, 2008
Усть-Каменогорск, 2008

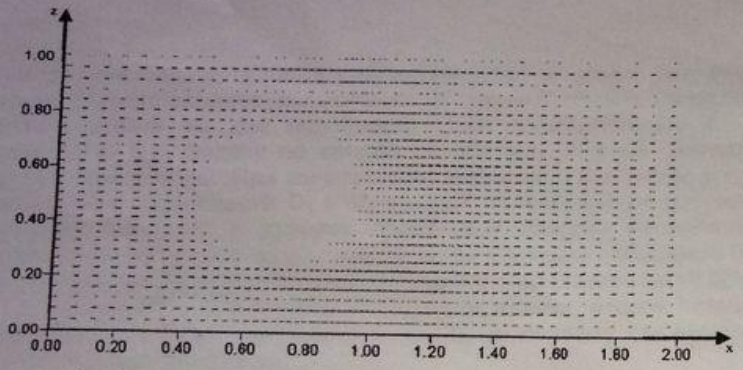


Рисунок 3 – Поле вектора скорости

На рисунке 3 приведены поле вектора скорости. Эти векторы направлены по касательной к линии тока. По результатам расчетов трехмерной модели, проводя линии тока, получается поверхность тока, заключающая внутри себя часть газа, называемая трубкой тока.

Рассмотрен метод Лакса-Вендроффа применительно к трехмерным задачам газовой динамики в многосвязных областях. Показана возможность использования метода для задач с взаимодействием разрывов, когда необходимо их выделение. Автоматически сгущаются узлы сетки к особенностям расположения препятствия. На примере тестовой задачи (обтекания шара) показана эффективность и применимость предлагаемого подхода.

Литература

1 Годунов С.К. Разностный метод численного расчета разрывных решений уравнений гидродинамики // Математический сборник. – 1959. – т. 47, № 3. – С. 271-306.

2 Самарский А.А., Попов Ю.П. Разностные методы решения задач газовой динамики. – М.: Наука, 1992. – 424 с.

3 Белоцерковский О.М., Андрущенко В.А., Шевелев Ю.Д. Динамика пространственных вихревых течений в неоднородной атмосфере. – М.: «Янус-К», 2000. – 456 с.

4 Шевелев Ю.Д., Сызранов Н.Г., Андрущенко В.А., Михалин В.А., Максимов Ф.А. Решение задач проектирования летательных аппаратов на многопроцессорных вычислительных комплексах // Математическое моделирование. – 2007. – т.19. – С. 25-38.

5 Бреславский П.В., Мажукин В.И. Моделирование взаимодействия ударных волн на динамически адаптирующихся сетках // Математическое моделирование. – 2007. – т.19, № 11. – С. 83-95.

6 Волков К.Н. Расчет свободного слоя смещения на основе метода моделирования крупных вихрей // Математическое моделирование. – 2007. – т.19, № 9. – С. 114-128.

7 Смагулов Ш.С., Данаев Н.Т., Темирбеков Н.М. Моделирование краевых условий для давления и полного напора в задачах гидродинамики с помощью метода фиктивных областей // Доклады Академии Наук России. – 2000. – т. 374, № 3 – С. 333-335.

УДК 004:518.9

Малгаждаров Е.А., Темирбеков А.Н.

ЧИСЛЕННАЯ РЕАЛИЗАЦИЯ ЭЛЛИПТИЧЕСКИХ УРАВНЕНИЙ В МНОГОПРОЦЕССОРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

С появлением высокопроизводительной техники стало возможным проводить моделирование весьма сложных явлений и процессов в механике, физике, экологии и в других отраслях науки. Основу модели составляют многомерные (пространственные) нестационарные задачи, исходная математическая формулировка которых весьма затруднительна. Решение таких проблем проводится в рамках численного эксперимента. При проведении систематического счета, для того, что бы получить более точные результаты и общей картины требовалось очень много машинного времени и оперативной памяти.

С появлением возможности проведения параллельного вычисления в локально соединенных ЭВМ-ах, стало актуальным использовать несколько компьютеров для сокращения машинного времени и экономия памяти машин.

Для этого разрабатывают специализированные параллельные алгоритмы и программы, реализующие эти алгоритмы. Более того, не все алгоритмы допускают распараллеливание, например, если алгоритм состоит в последовательной модификации одних и тех же данных, причем выбор ветки алгоритма шага интеграции зависит от исходных данных, то такой алгоритм очень сложно поддается распараллеливанию. Если имеются несколько альтернативных вариантов, возникает проблема выбора наиболее эффективного алгоритма.

Можно выделить два источника параллелизма в задаче. В первом случае имеются несколько независимых процессов обработки данных,

например вычисления и коммуникации. Во втором случае можно разделить сами данные на независимые части и производить их обработку (всю или некоторую ее часть) независимо [1]. Параллелизм по данным чаще встречается на практике. В численных методах решения уравнений в частных производных он возникает естественным образом в силу наличия вычислительной сетки, причем каждый ее узел или группа узлов являются кандидатами на данные, которые можно частично обрабатывать независимо.

Для проведения методических расчетов было реализовано параллельное вычисление алгоритма численного решения следующей задачи Дирихле для уравнения Пуассона.

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=g(x,y),\quad (x,y)\in\Omega$$

с граничным условием

$$u\big|_{\lambda}=0,$$

в области

$$\Omega=\{0\le x\le 1,\ 0\le y\le 1\}$$

с границей Г.

Построим равномерную сетку

$$\Omega_h=\{(x_i,y_j),\ i,j=\overline{1,n},\ h_1,h_2=1/n,\ x_i=(i-1)h_1,\ y_j=(j-1)h_2\}.$$

Запишем разностный аналог уравнений (1) в пятиточечном шаблоне:

$$\frac{y_{i+1,j}-2\cdot y_{i,j}+y_{i-1,j}}{h_1^2}+\frac{y_{i,j+1}-2\cdot y_{i,j}+y_{i,j-1}}{h_2^2}=g_{i,j},$$

со следующими граничными условиями:

$$y_{1,j}=y_{i,1}=y_{n_1,j}=y_{i,n_2}=0,\quad i,j=\overline{1,n}.$$

При $h\to 0 \Rightarrow y_{\bar{x}x}\to u_{xx}$, но тогда увеличится количество узлов и это может привести к нехватке оперативной памяти машины и увеличению машинного времени для вычисления задачи. Для массива типа float (вещественный тип) единица адресации (элемент массива) имеет размер 4 байта. В нашем случае для машины с оперативной памятью 256Мб размер массива не должен превышать $n_1\times n_2<6{,}7\cdot 10^7$. В противном случае возникает нехватка оперативной памяти. Поэтому

используем метод разделения области для параллельного вычисления.

Рассмотрим алгоритм метода разделения области:

В области Π_1 решаем задачу с условием Неймана на линии раздела областей γ : $x_1=h$

$$Lu_1^{k+1}=0,\ (x_1,x_2)\in\Pi_1$$

$$u_1^{k+1}=0\ \text{на}\ \partial\Pi_1\ (\partial\Pi_1=\Pi_1\setminus\gamma)$$

$$k_1\frac{\partial u_1^{k+1}}{\partial x_1}=k_2\frac{\partial u_2^{k+1}}{\partial x_2}\ \text{на}\ \gamma$$

В области Π_2 решаем задачу Дирихле, используя значение функции u_1^{k+1} на прямой $x_1=h$

$$Lu_2^{k+\frac{1}{2}}=0,\ (x_1,x_2)\in\Pi_2$$

$$u_2^{k+\frac{1}{2}}\Big|_{x_2=0,x_2=h}=0,\ u_2^{k+\frac{1}{2}}\Big|_{x_1=a}=V$$

$$u_2^{k+\frac{1}{2}}=u_1^{k+1}\ \text{на}\ \gamma,$$

Для численного решения задачи (4), (5) используется итерационный метод Зейделя [2]:

Для организации параллельного вычисления расчетная область была разделена на две подобласти (см. рис. 1). Счет проводился на двух машинах.

Выбрали функцию g(x,y) в следующем виде:

$$g(x,y)=2x(x-1)+2y(y-1)$$

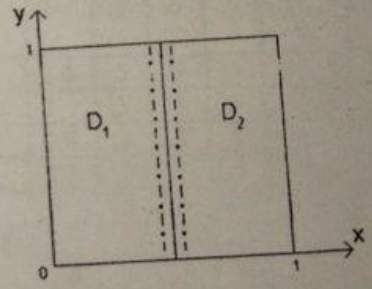


Рисунок 1 – Разделения расчетной области

В данной задаче область разделена 100x100 узлов. Для использование параллельного вычисления в данной задаче квадратную область делим на две подобласти (1-ая подобласть D_1 и 2-ая подобласть D_2). Задачу на каждой подобласти D_1 и D_2 вычисляют разные процессоры. При вычислениях значения подобласти в точках пересечений подобласти D_1 и D_2 каждый из процессору полученные значения передает к другому процессору. Задача параллельного вычисления считается эффективным когда передача сообщений будет минимальным. Значения в граничных точках передается не по одному, а массивами. После достижения точности задачи, второй процессор полученную матрицу подобласти D_2 передает первому для визуального представление результата.

Для того чтобы реализовать, программы распараллеливания создана программа с применением системы MPI.

Структура программы MPI

Каждая программа MPI содержит директиву препроцессора: #include "mpi.h"

Файл mpi.h содержит определения, макроопределения и прототипы функций, необходимых для компиляции программ MPI. Прежде чем вызывать любые другие функции MPI, нужно однократно вызвать функцию MPI_Init(). Ее аргументы – это указатели на параметры функции main() – argc и argv. Они позволяют системе выполнять любую специальную настройку, чтобы использовать библиотеку MPI. После того, как программа, использующая библиотеку MPI, закончилась, необходимо вызвать MPI_Finalize(). Эта функция завершает все незавершенные действия MPI - например, бесконечное ожидание передач. Типичная программа MPI имеет следующую структуру [3]:

```
#include "mpi.h"
    . . .
main(int argc, char** argv) {
    . . .
MPI_Init(&argc, &argv);
/* Функции MPI нельзя вызывать до этого момента */
    . . .
MPI_Finalize();
/* Функции MPI нельзя вызывать после этого момента */
    . . .
} /* main */
```

Описание основных функций

MPI_Comm_size(MPI_COMM_WORLD, &size)-переменную size

берется количество процессов. Процессы нумеруется 0...n.

MPI_Comm_rank(MPI_COMM_WORL,&rank)-переменную rank присваивается номер текущего процесса.

MPI_Send(u1s, n2, MPI_DOUBLE, 1, 0, MPI_COMM_WORLD); функция для передачи сообщения.

- u1s - имя передаваемого массива
- n2 - количество передаваемых элементов
- MPI_DOUBLE – тип передаваемых элементов
- 1 - номер процесса которому осуществляется передача данных
- 0 - тег сообщения
- MPI_COMM_WORLD - переключатель каналов

MPI_Recv(u1r, n2+1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &status); функция для получения сообщения.

- u1r – имя получаемого массива
- n2+1 – максимальное количество принимаемых элементов
- MPI_DOUBLE - тип принимаемых элементов
- 0 - номер передающего процесса
- 0 - тег сообщения
- MPI_COMM_WORLD – имя переключателя каналов
- &status – статус(состояние) передаваемых данных

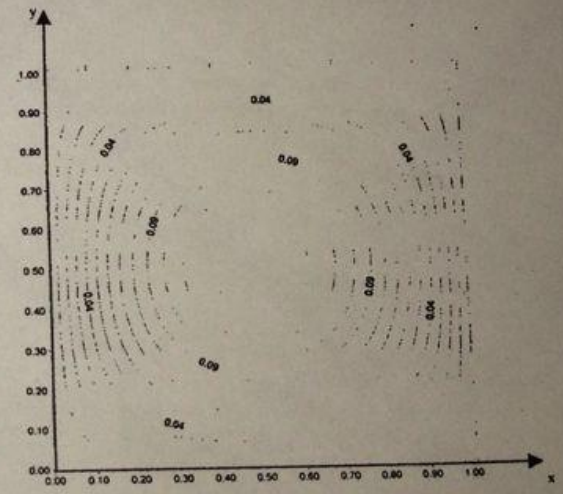


Рисунок 2 – Изолинии решения задачи