



Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра математической физики

Киселёв Евгений Иванович

Суперкомпьютерное моделирование и технологии

ОТЧЁТ О ВЫПОЛНЕНИИ ЗАДАНИЯ

Математическая постановка задачи

Рассмотрим задачу Дирихле для уравнения Пуассона:

$$\begin{cases} -\Delta u = -\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = f(x, y), & (x, y) \in D \in \mathbb{R}^2 \\ u(x, y) = 0, & (x, y) \in \gamma = \partial D \end{cases}$$

Пусть теперь $D \in \mathbb{I} = \{(x, y): A_x < x < B_x, A_y < y < B_y\}$, и $\bar{D}, \bar{\mathbb{I}}$ – замыкания областей D, \mathbb{I} . Пусть Γ – граница \mathbb{I} . $\hat{D} = \mathbb{I} \setminus \bar{D}$ назовём фиктивной областью. Выберем и зафиксируем малое $\varepsilon > 0$.

Рассмотрим задачу Дирихле в \mathbb{I} :

$$\begin{cases} -\frac{\partial}{\partial x} \left(k(x, y) \frac{\partial v}{\partial x} \right) - \frac{\partial}{\partial y} \left(k(x, y) \frac{\partial v}{\partial y} \right) = F(x, y), & (x, y) \in \gamma, \\ v(x, y) = 0, & (x, y) \in \Gamma \end{cases}$$

$$\text{где } k(x, y) = \begin{cases} 1, & (x, y) \in D \\ \frac{1}{\varepsilon}, & (x, y) \in \hat{D} \end{cases}, F(x, y) = \begin{cases} f(x, y), & (x, y) \in D \\ 0, & (x, y) \in \hat{D} \end{cases}$$

Требуется найти непрерывную в $\bar{\mathbb{I}}$ функцию $v(x, y)$, являющуюся решением данной задачи, такую, что

$$\lim_{\substack{(x, y) \rightarrow (x_0, y_0) \\ (x, y) \in D}} (W(x, y), n(x_0, y_0)) = \lim_{\substack{(x, y) \rightarrow (x_0, y_0) \\ (x, y) \in \hat{D}}} (W(x, y), n(x_0, y_0)),$$

где $W(x, y) = -k(x, y) \left(\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y} \right)$, $n(x, y)$ – вектор единичной нормали к границе γ в точке (x, y) , определённый всюду или почти всюду на кривой.

Также известно, что $\max_{(x, y) \in D} |v(x, y) - u(x, y)| < C\varepsilon$, $C > 0$.

Разностная схема и метод скорейшего спуска

Краевую задачу предлагается решать численно методом конечных разностей.

Введём равномерную прямоугольную сетку $\bar{\omega}_h = \bar{\omega}_x \times \bar{\omega}_y$

$$\bar{\omega}_x = \{x_i = A_x + ih_x, i = \overline{0, M}\}, \bar{\omega}_y = \{y_j = A_y + jh_y, j = \overline{0, N}\},$$

$$\text{где } h_x = \frac{B_x - A_x}{M}, h_y = \frac{B_y - A_y}{N}.$$

Обозначим за ω_h внутренние узлы сетки. Рассмотрим линейное пространство H функций, заданных на ω_h , и пусть ω_{ij} – значение функции из H в $(x_i, y_j) \in \omega_h$. Будем считать, что в H задано скалярное произведение и евклидова норма: $(u, v) = \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} h_x h_y u_{ij} v_{ij}$, $\|u\|_E = \sqrt{(u, u)}$.

В методе конечных разностей дифференциальная задача математической физики заменяется конечно-разностной операторной задачей вида $A\omega = B$, где $A: H \rightarrow H$, $B \in H$. Такая задача называется разностной схемой. Решение этой задачи считается численным решением исходной дифференциальной задачи.

Дифференциальное уравнение во всех точках ω_h аппроксимируется следующим разностным уравнением:

$$-\frac{1}{h_x} \left(a_{i+1j} \frac{\omega_{i+1j} - \omega_{ij}}{h_x} - a_{ij} \frac{\omega_{ij} - \omega_{i-1j}}{h_x} \right) - \frac{1}{h_y} \left(b_{ij+1} \frac{\omega_{ij+1} - \omega_{ij}}{h_y} - b_{ij} \frac{\omega_{ij} - \omega_{ij-1}}{h_y} \right) = F_{ij},$$

$$i = \overline{1, M-1}, j = \overline{1, N-1},$$

$$\text{где } a_{ij} = \frac{1}{h_y} \int_{y_{j-1/2}}^{y_{j+1/2}} k(x_{i-1/2}, t) dt, b_{ij} = \frac{1}{h_x} \int_{x_{i-1/2}}^{x_{i+1/2}} k(t, y_{j-1/2}) dt,$$

$$i = \overline{1, M}, j = \overline{1, N},$$

$$x_{i\pm 1/2} = x_i \pm \frac{h_x}{2}, y_{j\pm 1/2} = y_j \pm \frac{h_y}{2}$$

$$\text{и } F_{ij} = \frac{1}{h_x h_y} \iint_{\mathbb{I}_{ij}} F(x, y) dx dy, i = \overline{1, M-1}, j = \overline{1, N-1},$$

$$\text{где } \mathbb{I}_{ij} = \{(x, y): x_{i-1/2} \leq x \leq x_{i+1/2}, y_{j-1/2} \leq y \leq y_{j+1/2}\}$$

Краевые условия аппроксимируется точным равенством $\omega_{ij} = \omega(x_i, y_j) = 0$ для $(x_i, y_j) \in \Gamma$. Так как эти переменные исключаются из общей системы уравнений, то количество неизвестных совпадает с числом

уравнений; при этом система является линейной и, следовательно, имеет единственное решение.

Интегралы в равенствах выше можно упростить.

$$a_{ij} = \begin{cases} 1, & [(x_{i-1/2}, y_{j-1/2}), (x_{i-1/2}, y_{j+1/2})] \in D \\ \frac{1}{\varepsilon}, & [(x_{i-1/2}, y_{j-1/2}), (x_{i-1/2}, y_{j+1/2})] \in \widehat{D} \\ \frac{l_{ij}}{h_y} + \frac{\left(1 - \frac{l_{ij}}{h_y}\right)}{\varepsilon}, & \text{иначе } (l_{ij} - \text{длина отрезка} \in D) \end{cases}$$

$$b_{ij} = \begin{cases} 1, & [(x_{i-1/2}, y_{j-1/2}), (x_{i+1/2}, y_{j-1/2})] \in D \\ \frac{1}{\varepsilon}, & [(x_{i-1/2}, y_{j-1/2}), (x_{i+1/2}, y_{j-1/2})] \in \widehat{D} \\ \frac{l_{ij}}{h_x} + \frac{\left(1 - \frac{l_{ij}}{h_x}\right)}{\varepsilon}, & \text{иначе } (l_{ij} - \text{длина отрезка} \in D) \end{cases}$$

$$F_{ij} = \begin{cases} f(x_i, y_j), & \mathbb{I}_{ij} \subset D \\ 0, & \mathbb{I}_{ij} \subset \widehat{D} \\ \frac{S_{ij}f(x_i^*, y_j^*)}{h_x h_y}, & \text{иначе } (S_{ij} - \text{площадь } \mathbb{I}_{ij} \cap D, (x_i^*, y_j^*) \in \mathbb{I}_{ij} \cap D) \end{cases}$$

Приближённое решение разностной схемы ищется методом скорейшего спуска:

$$\omega_{ij}^{k+1} = \omega_{ij}^k - \tau_{k+1} r_{ij}^k,$$

где $r_{ij}^k = A\omega_{ij}^k - F_{ij}$ – невязка, $\tau_{k+1} = \frac{(r^k, r^k)}{(Ar^k, r^k)}$ – итерационный параметр.

Критерий остановки этого метода выглядит следующим образом:

$$\|\omega_{ij}^{k+1} - \omega_{ij}^k\|_E < \delta, \text{ где } \delta > 0$$

Замечание: в численной схеме указано взять $\varepsilon = \left(\max(h_x, h_y)\right)^2$.

Программная реализация

Этапы выполнения программы:

- 1) Инициализация данных
- 2) Заполнение матриц a_{ij} , b_{ij} , F_{ij}
- 3) Цикл while (условие: пока $\|\omega_{ij}^{k+1} - \omega_{ij}^k\|_E < \delta$). В цикле
 - a. Вычисление r_{ij}^k
 - b. Вычисление τ_{k+1}
 - c. Вычисление ω_{ij}^{k+1}
 - d. Вычисление $\|\omega_{ij}^{k+1} - \omega_{ij}^k\|_E$
 - e. В случае невыполнения условия цикла: копирование ω_{ij}^{k+1} в ω_{ij}^k
- 4) Сохранение решение
- 5) Освобождение памяти

Пункты 2, 3(а-е) задают циклы по сетке. Их можно распараллелить средствами OpenMP. Также в задании нужно было провести распараллеливание с помощью технологии MPI. В программе с помощью MPI_Dims_create выбирается наилучшее разбиение доступного числа процессоров в двумерную сетку. Инициализация a_{ij} , b_{ij} , F_{ij} происходит в локальных матрицах, размеры которых определяет сетка по процессорам. Расчёты для последовательной реализации (1 процессор и 1 нить) приведены для вариантов OpenMP и MPI+OpenMP соответственно, так как для компиляции использовались разные варианты компиляторов (IBM и mpicc++ соответственно) и время работы отличалось:

Реализация	Число точек сетки $N \times M$	Число итераций	Время решения
OpenMP	90×80	13223	4.675[s]
OpenMP	180×160	8817	12.587[s]
MPI+OpenMP	90×80	13223	8.13[s]
MPI+OpenMP	180×160	8817	24.38[s]

График решения на сетке (180, 160) для $f(x, y) = 1$:

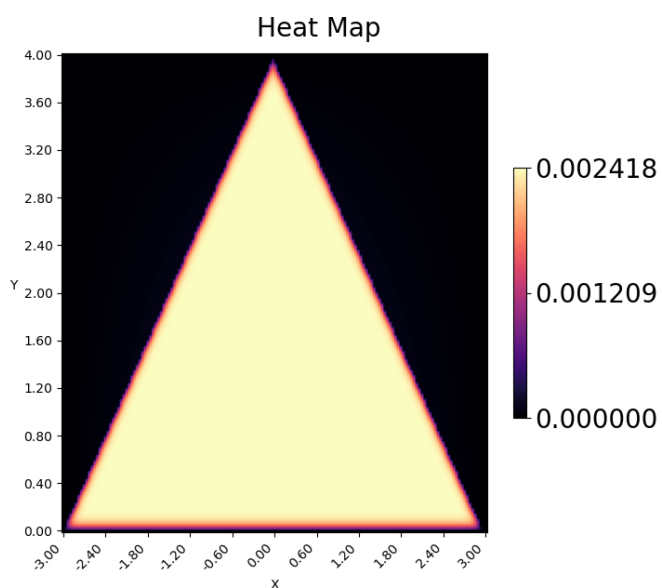


Таблица 1:

Количество нитей OpenMP	Число точек сетки $N \times M$	Число итераций	Время решения	Ускорение
2	90×80	13223	2.461[s]	1.9
4	90×80	13223	1.392[s]	3.36
8	90×80	13223	1.114[s]	4.2
16	90×80	13223	0.855[s]	5.46
4	180×160	8817	3.288[s]	3.83
8	180×160	8817	2.23[s]	5.64
16	180×160	8817	1.956[s]	6.43
32	180×160	8817	1.876[s]	6.7

Таблица 2:

Количество процессоров MPI	Количество нитей OpenMP	Число точек сетки $N \times M$	Число итераций	Время решения	Ускорение
2	1	90×80	13233	4.41[s]	1.84
2	2	90×80	13233	4.186[s]	1.94
2	4	90×80	13233	2.362[s]	3.14
2	8	90×80	13233	1.658[s]	4.9
4	1	180×160	8817	5.93[s]	4.1
4	2	180×160	8817	5.838[s]	4.17
4	4	180×160	8817	4.68[s]	5.2
4	8	180×160	8817	2.997[s]	8.16

Заключение

Таким образом была проделана работа по реализации метода конечных разностей в том числе и с помощью технологий OpenMP и MPI. В целом можно заключить, что с ростом числа процессоров/нитей ускорение программы увеличивается. Важно заметить, что времена работы программ, скомпилированных разными компиляторами, отличаются, вероятно из-за того, что компилятор IBM лучше подходит для железа распределённого кластера Полус. Также можно заметить, что поведение ускорения программы имеет схожую динамику для случаев OpenMP и MPI+OpenMP реализаций (например, в случае OpenMP ускорение не сильно отличается для случаев 2-х и 4-х нитей, а у MPI+OpenMP реализации аналогичное поведение наблюдается для случаев 1-й и 2-х нитей).