Отчёт по курсу «Суперкомпьютерное моделирование и технологии» для студентов первого и второго потоков, вариант 7, Жэнь Цзежуй 618-1

1 Математическая постановка задачи

Требуется приближённо решить двумерную задачу Дирихле для уравнения Пуассона в области сложной формы — так называемой «области-сапожок»:

$$-\Delta u(x,y) = 1, \quad (x,y) \in D, \qquad u|_{\partial D} = 0.$$

Область D определяется как разность двух квадратов:

$$D = \{(x,y) : -1 < x < 1,\ -1 < y < 1\} \setminus \{(x,y) : 0 < x < 1,\ 0 < y < 1\}.$$

То есть из квадрата $(-1,1)\times(-1,1)$ удалён его правый верхний квадрат $(0,1)\times(0,1)$. Таким образом граница области имеет Г-образный вид, напоминающий сапог.

На границе задаётся условие Дирихле $u = 0$.

2 Численный метод решения

Для решения используется **метод фиктивных областей**. Вне D вводится малая проницаемость $1/\varepsilon$, что обеспечивает приближение граничного условия $u = 0$. В прямоугольнике $\Pi = [-1,1]\times[-1,1]$ решается модифицированное уравнение

$$-\nabla\cdot\big(k(x,y)\nabla v\big) = F(x,y), \qquad v\big|_{\Gamma} = 0,$$

где

$$k(x,y) = \begin{cases} 1, & (x,y) \in D, \\ 1/\varepsilon, & (x,y) \notin D, \end{cases} \qquad F(x,y) = \begin{cases} 1, & (x,y) \in D, \\ 0, & (x,y) \notin D. \end{cases}$$

Параметр $\varepsilon = h^2$ обеспечивает выполнение граничного условия на криволинейной границе.

На равномерной сетке

$$x_i = -1 + ih, \qquad y_j = -1 + jh, \qquad i,j = 0..N, \qquad h = \frac{2}{N},$$

применяется аппроксимация второго порядка:

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} = 1.$$

На узлах вне области D значение функции принимается равным нулю. Полученная система $A\,w = B$ является разреженной, симметричной и положительно определённой.

3 Последовательный код программы (Задание 1)

В последовательной версии для решения разреженной симметричной положительно определённой системы

$$Aw = f,$$

полученной после дискретизации уравнения Пуассона в области сложной формы, используется **метод сопряжённых градиентов**. Матрица A не хранится явно; оператор применяется «на лету» с использованием пятиточечного шаблона.

Начальное приближение выбирается как $w^{(0)} = 0$. На каждой итерации вычисляются следующие величины:

$$r^{(k)} = f - Aw^{(k)}, \qquad p^{(k)} = r^{(k)} + \beta^{(k-1)} p^{(k-1)},$$

$$\alpha^{(k)} = \frac{(r^{(k)}, r^{(k)})_E}{(p^{(k)}, Ap^{(k)})_E}, \qquad w^{(k+1)} = w^{(k)} + \alpha^{(k)} p^{(k)},$$

$$r^{(k+1)} = r^{(k)} - \alpha^{(k)} Ap^{(k)}, \qquad \beta^{(k)} = \frac{(r^{(k+1)}, r^{(k+1)})_E}{(r^{(k)}, r^{(k)})_E}.$$

Итерационный процесс завершается, когда энергетическая норма невязки удовлетворяет условию

$$\|r^{(k)}\|_E < 10^{-6}.$$

Компиляция и запуск:

```
g++ -O3 -std=c++11 task1.cpp -o task1
./task1
```

Результаты:

Расчёты выполнены на сетках 10×10, 20×20 и 40×40. Получены следующие значения количества итераций, невязки и времени работы:

M	N	Итераций	Остаток	Время (s)
10	10	132	9.5789×10^{-7}	0.000588
20	20	572	9.7168×10^{-7}	0.004901
40	40	2353	9.9796×10^{-7}	0.059938

Метод сопряжённых градиентов существенно ускоряет сходимость по сравнению с методом скорейшего спуска: рост числа итераций соответствует теоретической оценке $O(\sqrt{N})$ для задач типа Пуассона.

4 OpenMP

4.1 Задание 2

В рамках задания 2 был реализован численный решатель для двумерного уравнения Пуассона на области-сапожке.

4.1.1 Используемый метод

Алгоритм относится к классу предобусловленных методов сопряженных градиентов и включает следующие этапы:

- вычисление остатка $r = F - Aw$;
- применение диагонального предобуславливания

$$z = D^{-1}r,$$

  где D — диагональная часть оператора A;
- вычисление шага

$$\alpha = \frac{(r, z)}{(p, Ap)};$$

- обновление решения и остатка;
- обновление направления поиска

$$p_{k+1} = z_{k+1} + \beta_k p_k, \qquad \beta_k = \frac{(r_{k+1}, z_{k+1})}{(r_k, z_k)}.$$

Использование диагонального предобуславливания Якоби улучшает спектральные свойства матрицы дискретизации и ускоряет сходимость по сравнению с обычным методом сопряженных градиентов.

4.1.2 Результаты расчёта для сетки 40×40

Threads	TOL	Количество итераций	Остаток	Время (s)
1	1e−6	3028	9.99433×10^{-7}	1.01529
4	1e−6	3028	9.99433×10^{-7}	0.356572
16	1e−6	3028	9.99433×10^{-7}	0.258314

4.2 Задание 3

Цель. Повысить точность и ускорить сходимость за счёт использования потоковой (физически корректной) дискретизации и решателя **метода сопряжённых градиентов** с предобуславливанием Якоби.

В задаче `task3` коэффициенты $a_{i\pm1/2,j}$ и $b_{i,j\pm1/2}$ учитывают долю ячеек, принадлежащей области D, что обеспечивает точное соблюдение граничных условий и симметрию оператора.

4.2.1 Основные фрагменты

Предобуславливание и скалярное произведение:

```
for (int i=1; i<M; ++i)
for (int j=1; j<N; ++j)
    z[i][j] = r[i][j] / diag[i][j];

```

```
double rz = 0.0;
for (int i=1; i<M; ++i)
for (int j=1; j<N; ++j)
    rz += r[i][j] * z[i][j];
```

Вычисление α и обновление w, r:

```
applyA(p, Ap);
double pAp = dotE(p, Ap);
double alpha = rz / pAp;

for (int i=1; i<M; ++i)
for (int j=1; j<N; ++j){
    w[i][j] += alpha * p[i][j];
    r[i][j] -= alpha * Ap[i][j];
}
```

Вычисление β и обновление направления p:

```
applyDinv(r, z);
double rz_new = dotE(r, z);
double beta = rz_new / rz;

for (int i=1; i<M; ++i)
for (int j=1; j<N; ++j)
    p[i][j] = z[i][j] + beta * p[i][j];

rz = rz_new;
```

4.2.2 Компиляция и запуск

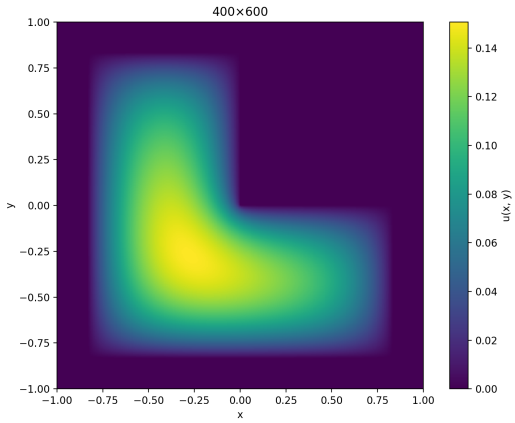
```
g++ -o3 -std=c++11 -fopenmp task3.cpp -o task3
./task3
```

4.2.3 Визуализация

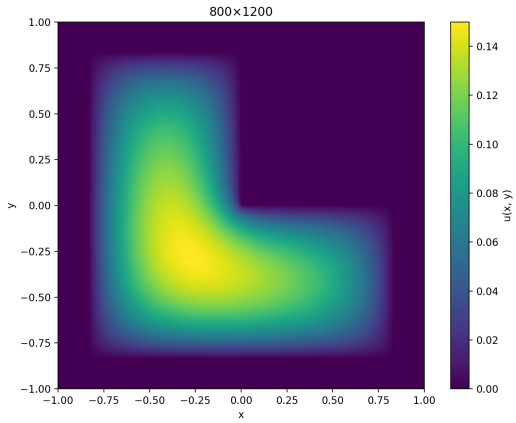
На Рис. 1 видно, что максимум функции $u(x, y)$ расположен внутри области, а на границе значение строго равно нулю. Более мелкая сетка даёт более гладкий профиль решения.

4.2.4 Производительность и ускорение

Число потоков	M×N	Итераций	Время (s)	Ускорение
1	400×600	2749	256.698221	1.00
1	800×1200	5237	1948.584467	1.00



(a) Тепловая карта для сетки 400×600

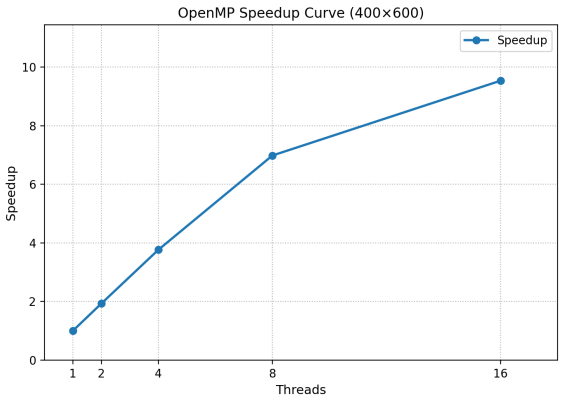


(b) Тепловая карта для сетки 800×1200

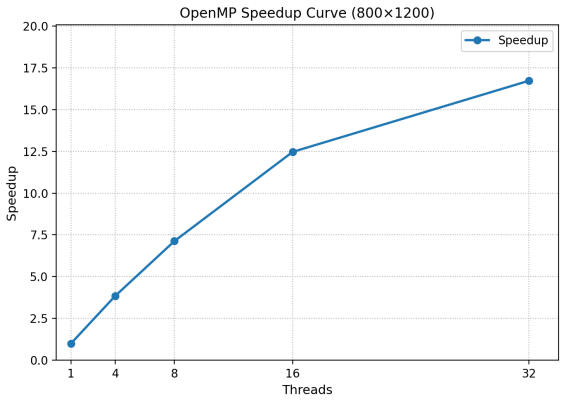
Рис. 1: Тепловая карта OpenMP

Число потоков	M×N	Итераций	Время (s)	Ускорение
2	400×600	2749	132.982183	1.93
4	400×600	2749	68.166429	3.77
8	400×600	2749	36.760456	6.99
16	400×600	2749	26.923483	9.54
4	800×1200	5237	505.865782	3.852
8	800×1200	5237	273.394149	7.127
16	800×1200	5237	156.307519	12.473
32	800×1200	5237	116.491348	16.731

Таблица 1: Таблица с результатами расчетов на ПВС IBM Polus (OpenMP код).



(a) 400 × 600



(b) 800 × 1200

Рис. 2: Графики ускорения OpenMP