Лабораторная работа №4

Дискретизация расчетной области

- Расчетная область [0, 1] x [0, 1] покрывается прямоугольной сеткой с постоянным шагом: nx точек по оси OX и ny точек по оси OY
- Расчетная сетка – массив [ny, nx] чисел (температура)
- Переход от индекса ячейки [i, j] к координатам в области [0, 1] x [0, 1]:

  x = j * 1.0 / (nx – 1.0)

  y = i * 1.0 / (ny – 1.0)

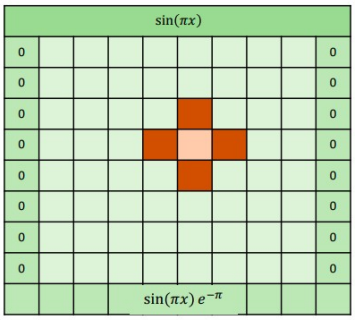
Разностная аппроксимация

- Вторые производные аппроксимируются на расчетной сетке разностным уравнением с применением четырехточечного шаблона

$$\Delta U = \frac{d^2U}{dx^2} + \frac{d^2U}{dy^2} = 0$$

- Новое значение в каждой точке сетки равно среднему из предыдущих значений четырех ее соседних точек (схема «крест»)

```
grid_new[i ,j] = (grid[i - 1, j] + grid[i, j + 1] +
                  grid[i + 1, j] + grid[i, j - 1]) / 4
```



Суть распараллеливания

Двумерная (2D) декомпозиция расчётной области на прямоугольные области

- Каждому процессу назначается подмассив [ny / p, nx / p] строк расчетной сетки
- Сетка хранится в памяти в распределенном виде
- **Проблема** – для расчета значений некоторых точек требуются данные соседних полос, которые находятся в памяти других процессов

Теневые ячейки (halo, ghost cells) для хранения значений из соседних процессов

Неблокирующие функции Send/Recv. Возврат из функции происходит сразу после инициализации процесса передачи/приема. Буфер использовать нельзя до завершения операции

MPI_Irecv() - прием

int MPI_Irecv(void* buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Request *request)

OUT buf - адрес для принимаемых данных;
IN count - максимальное число принимаемых элементов;
IN datatype - тип элементов принимаемого сообщения;
IN source - номер процесса-отправителя;
IN tag - идентификатор сообщения;
IN comm- коммуникатор;
OUT request - "запрос обмена".

MPI_Isend() - передача

int MPI_Isend(void* buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm, MPI_Request *request)

IN buf - адрес начала расположения передаваемых данных;
IN count - число посылаемых элементов;
IN datatype - тип посылаемых элементов;
IN dest - номер процесса-получателя;
IN tag - идентификатор сообщения;
IN comm- коммуникатор;
OUT request - "запрос обмена".

all_to_all — это функция, в рамках которой Каждый процесс передает свое сообщение всем процессам

Блокирующее ожидание завершения операции

MPI_Waitall() - Ожидает завершения всех заданных MPI-запросов.

int MPI_Waitall(int count, MPI_Request array_of_requests[], MPI_Status array_of_statuses[])
- принимает

count - длина списка (целое число)

array_of_requests - массив дескрипторов запросов (массив дескрипторов) - возвращает

array_of_statuses - массив статусных объектов (массив Статусов). Может быть `MPI_STATUSES_IGNORE`.