МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. Ломоносова
Факультет Вычислительной Математики и Кибернетики

Практикум по учебному курсу
"РАСПРЕДЕЛЁННЫЕ СИСТЕМЫ"

ОТЧЕТ

о выполненном задании

Реализация MPI_ALLREDUCE

студентки 428 учебной группы факультета ВМК МГУ
Шелепнёвой Дарьи Дмитриевны

Москва, 2022г.

Содержание

1 Постановка задачи

В транспьютерной матрице размером 8*8, в каждом узле которой находится один процесс, необходимо выполнить операцию нахождения максимума среди 64 чисел (каждый процесс имеет свое число). Найденное максимальное значение должно быть получено на каждом процессе.

Реализовать программу, моделирующую выполнение операции $MPI\_ALLREDUCE$ на транспьютерной матрице при помощи пересылок MPI типа точка-точка.

Оценить сколько времени потребуется для выполнения операции $MPI\_ALLREDUCE$, если все процессы выдали эту операцию редукции одновременно. Время старта равно 100, время передачи байта равно 1 (Ts=100,Tb=1). Процессорные операции, включая чтение из памяти и запись в память, считаются бесконечно быстрыми.

2 Алгоритм

2.1 Описание алгоритма

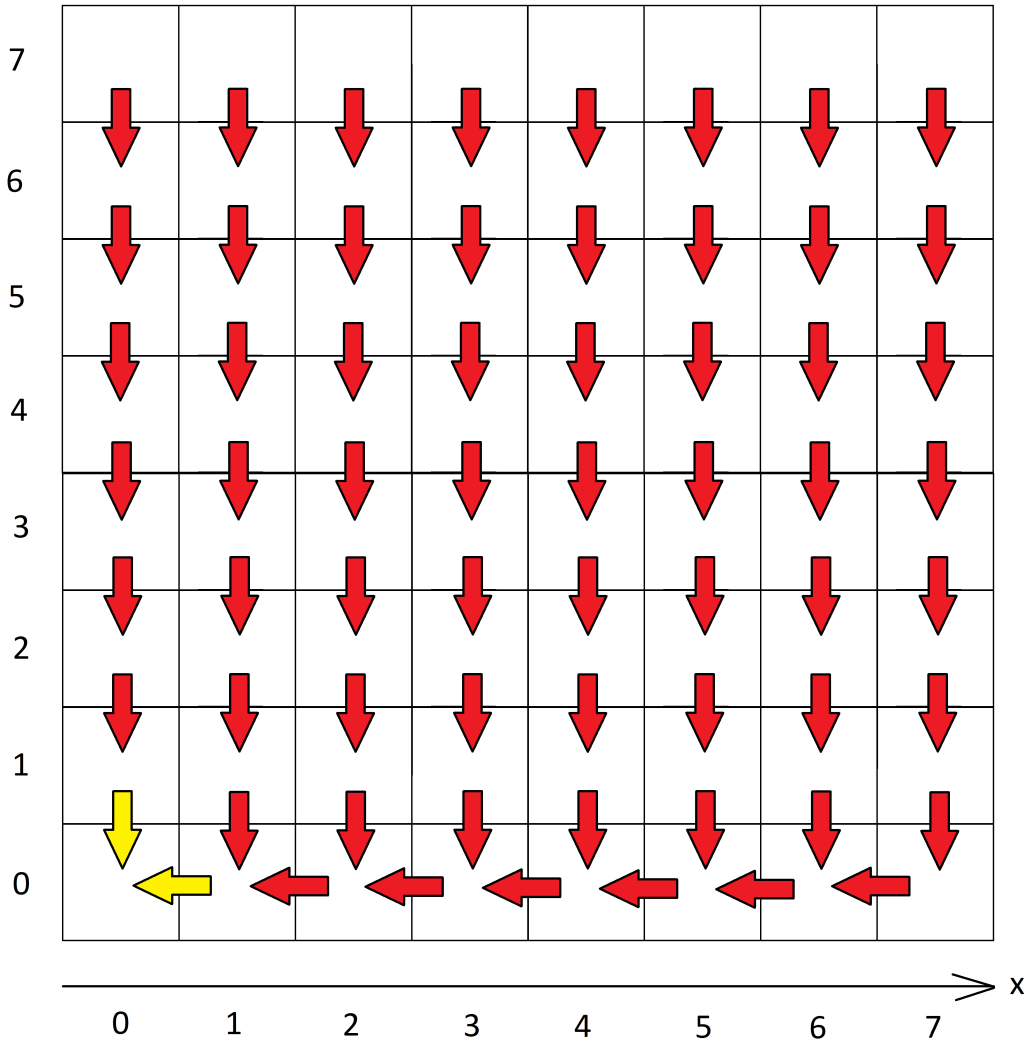

Рис. 1: Сбор максимума в одной ячейке

Рассмотрим матрицу $8 * 8$, нижняя левая клетка имеет индекс $(0, 0)$, верхняя правая индексируется $(7, 7)$. Наибольшее число будем находить на процессе $(0, 0)$ и затем отправлять его всем остальным процессам. Нахождение максимума на процессе $(0, 0)$ представлено на Рис. 1. После того, как результат максимума будет лежать в ячейке $(0, 0)$, мы отправляем его обратно (фактически производим инверсию стрелок) по всем процессам, как показано на Рис. 2.

Разобьем полученный алгоритм на шаги, которые между мобой аналогичны (от-

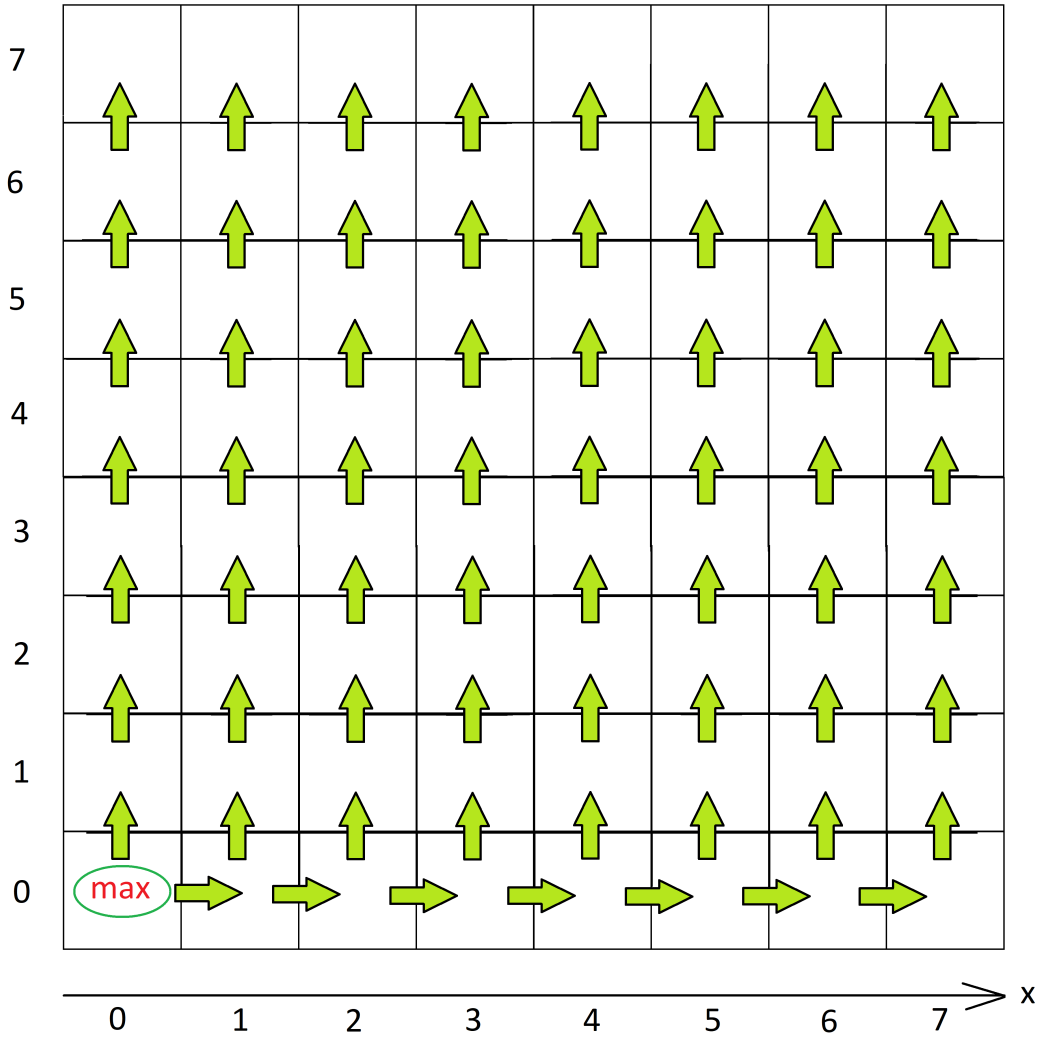

Рис. 2: Рассылка максимума по ячейкам

носительно стрелок на рисунках), далее эти шаги преобразуем в функции:

- "Вперед вниз"
  Процессы, находящиеся в координатах (x, 7) (где $x = \{0, 1, 2, 3, 4, 5, 6, 7\}$), отправляют свое число процессам, находящимся в координатах (x, 6). Эти процессы получают число, сравнивают его со своим и отправляют наибольшее из них процессам, находящимся в координатах (x, 5). Эти процессы аналогично отправляют наибольшее число процессам, находящимся в координатах (x, 4) и так далее.

- "Вперед влево"
  Процесс, находящийся в координатах (7, 0), получает число от процесса с коор-

динатами (7, 1), сравнивает со своим и отправляет наибольшее из них в процесс с координатами (6, 0). Процесс, находящийся в координатах (6, 0), получает числа от процессов с координатами (7, 0) и (6, 1), сравнивает их со своим и отправляет наибольшее процессу (5, 0). Процесс с координатами (5, 0) аналогично отправляет наибольшее число процессу (4, 0) и так далее.

- "Определение максимума среди 64 чисел"
  Процесс (0,0) получает числа от процессов с координатами (0,1) и (1,0), сравнивает их со своим и находим наибольшее. Это число и будет максимумом среди 16 чисел. Теперь нужно разослать это число оставшимся процессам.

- "Обратно вправо"
  Из (0,0) полученное число отправляется к (1, 0), оттуда в (2,0) оттуда в (3,0) и так далее до (7, 0).

- "Обратно вверх"
  Из (x, 0) полученное число отправляется вверх, то есть в (x, 1) соответственно. От этих процессов число попадает в (x, 2) соответственно. Итак далее до (x, 7) соответственно.

2.2 Реализация алгоритма

Код можно найти на гитхабе: `https://github.com/DariaShel/skipod`

2.3 Инструкции по запуску

Для запуска необходимо выполнить следующие команды в терминале:

- **mpicc mpi_allreduce.c -o mpi_allreduce**
- **mpiexec -np 16 mpi_allreduce —oversubscribe —with-ft ulfm**

3 Временная оценка

Дольше всех будет идти сообщение от (7, 7) до (0, 0). Ему нужно преодолеть 14 пересылок в одну сторону и столько же обратно. На каждую пересылку нужно потратить $Ts + 4 * Tb$, где $Ts = 100, Tb = 1, (4 = sizeof(int))$.

Значит в итоге получим:

$T = 2 * 14 * (Ts + 4 * Tb) = 2912$