Домашнее задание 2

Задача 1. Метод *k*-средних (*k*-means)

Постановка задачи

Реализуйте параллельную версию метода *k*-средних с использованием технологии OpenMP, взяв за основу готовый последовательный вариант программы. Обоснуйте в отчете выбранную стратегию распараллеливания.

Постройте графики зависимости коэффициентов ускорения *S* и эффективности *E* параллельной реализации от количества потоков (от 2 до 8) при фиксированном размере задачи и количестве кластеров. Объясните полученные результаты. Определите оптимальное количество потоков для выполнения программы.

Измерьте ускорение параллельной реализации при «оптимальном» количестве потоков

- в зависимости от количества точек, при фиксированном количестве кластеров
- в зависимости от количества кластеров (начиная с 2), при фиксированном количестве точек

Объясните полученные результаты.

Входные данные

Для генерации входных данных для метода *k*-средних предоставляется программа `data-gen.cpp`.

Формат запуска программы:

```
$ data-gen <размерность> <количество точек> <количество кластеров> \
           <выходной файл >
```

При решении можно всегда генерировать данные одной размерности (например, 2).

Последовательная реализация

Последовательная реализация метода *k*-средних находится в файле kmeans.cpp.

Формат запуска программы:

```
$ kmeans <количество кластеров> <входной файл> <выходной файл>
```

Параллельная реализация

Параллельная реализация должна иметь точно такой же набор аргументов и использовать те же форматы данных, что и последовательная. Количество потоков должно передаваться через переменную среды окружения OMP_NUM_THREADS.

Программа должна быть написана на C++.

Замеры времени

Все требуемые замеры следует проводить на учебном кластере. Запуск OpenMP-программы должен осуществляться на одном из узлов кластера в монопольном режиме (#PBS -l nodes=1:ppn=8).

При вычислении коэффициентов ускорения и эффективности всегда следует измерять полное время выполнения программ, включая ввод-вывод и инициализацию.

Задача 2. Игра «Жизнь» (Conway's Game of Life)

Постановка задачи

Реализуйте параллельную версию игры «Жизнь» (Conway's Game of Life) с использованием стандарта MPI. В отчете обоснуйте выбранную стратегию распараллеливания.

Постройте графики зависимости коэффициентов ускорения *S* и эффективности *E* параллельной реализации от:

- количества процессов (от 2 до 64)
- размера игрового поля
- количества итераций

Входные данные

Игровое поле имеет размер *N*x*N* (количество строк равно количеству столбцов). Считается, что верхняя граница поля «соединена» с нижней, а левая граница — с правой (зацикленное, бесконечное игровое поле).

Для генерации начальной конфигурации поля предоставляется программа `data-gen.c`.

Формат запуска программы:

```
$ data-gen <N> <выходной файл>
```

Каждая строка поля размещается в отдельной строке входного файла. Живые клетки

обозначаются символом «X», а мертвые - «.».

Последовательная реализация

Предоставляется две последовательных реализации:

- life.c — простейший алгоритм расчета нового поколения
- life2.c — алгоритм, просматривающий только ранее изменившиеся клетки и их соседей

Формат запуска программ одинаковый:

```
$ life <N> <входной файл> <количество итераций> <выходной файл >
```

После окончания программы выходной файл содержит конфигурацию поля после заданного количества итераций в таком же формате, что и входной файл.

Параллельная реализация

Параллельная реализация должна иметь точно такой же набор аргументов и использовать те же форматы данных, что и последовательная.

Программа должна быть написана на C или C++.

Замеры времени

Все требуемые замеры следует проводить на учебном кластере. Запуск программы должен осуществляться на одном или нескольких узлах кластера через систему очередей (см. на сайте инструкции по работе с кластером).

При вычислении ускорения и эффективности всегда следует измерять полные времена выполнения программ, включая ввод-вывод.