|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **«Программное обеспечение ЭВМ и информационные технологии» (ИУ7)**

**Домашнее задание**

**Дисциплина:** Программирование параллельных процессов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-31М |  |  | В.Д. Коноваленко |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | А.П. Ковтушенко |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Постановка задачи**

Дано N прямоугольников со сторонами ориентированными параллельно осям. Прямоугольники частично перекрываются. Периметром называется общая длина внутренних и внешних границ фигуры, полученной наложением всех прямоугольников. Найти периметр.

Обосновать проектное решение (выбор алгоритма). Обеспечить равномерную загрузку процессоров. Результат вывести в текстовый файл. Исследовать зависимость времени счета от размерности задачи и количества процессоров..

**Способ реализации**

Алгоритм решения данной задачи:

1. Рассматривать всю область задачи скользящим окном по возрастанию вдоль оси ОX.
2. Сформировать массив, где каждому элементу соответствует единичный отрезок OY. Значением массива является количество прямоугольников, находящихся на каждом единичном отрезке.
3. Если встретилась грань «начала» прямоугольнкиа (соответствует минимальной координате x прямоугольника), то прибавить 1 к каждому соответствующему элементу массива из п.2. В случае, если предыдущее значение элемента было равно 0, значит данный отрезок является частью внешнего периметра и его необходимо прибавить к итоговому периметру.
4. Если встретилась грань «конца» прямоугольника (соответствует максимальной координате х прямоугольника) , то отнять 1 от каждого соответствующего элемента массива из п.2. В случае, если предыдущее значение элемента было равно 1, значит данный отрезок является частью внешнего периметра и его необходимо прибавить к итоговому периметру.
5. На каждом шаге необходимо вычислять изменение по оси ОХ от предыдущего шага и число внешних горизонтальных граней прямоугольников (при помощи массива из п.2), перемноженное значение прибавить к итоговому периметру.

С использованием MPI данный алгоритм можно реализовать, изменив 1 шаг. Всю область необходимо разделить на вертикальные фрагменты, периметр на которых будет вычисляться разными узлами параллельно. В таком случае 0 узел должен разослать всем другим узлам список прямоугольников, после чего каждый из узлов должен обратиться к нулевому и запросить фрагмент для вычисления. Вычислив периметр на данном фрагменте, узел отправляет нулевому узлу результат и запрашивает следующий. Нулевой узел, в свою очередь, вычисляет итоговый периметр фигуры путём сложения всех результатов вычислений других узлов.

**Результаты**

Были проведены замеры времени выполнения алгоритма для разного количества прямоугольников: 50, 100, 1 000, 10 000, 100 000, 1 000 000 и разного количества узлов: 1, 2, 4, 8, 10. Результаты в секундах представлен в таблице ниже:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Кол-во узлов / кол-во прямоугольников | 1 | 2 | 4 | 8 | 10 |
| 50 | 0,378512 | 0,759791 | 0,417859 | 0,445513 | 0,478217 |
| 100 | 0,727823 | 0,923788 | 0,815514 | 0.866344 | 0,890208 |
| 1000 | 0,968508 | 0,992154 | 1,010982 | 1,03979 | 0,192762 |
| 10000 | 1,004739 | 0,706083 | 0,492919 | 0,406771 | 0,406414 |
| 100000 | 7,996604 | 5,018568 | 2,640238 | 1.546228 | 1,217692 |
| 1000000 | 60,032183 | 44,578349 | 18,95778 | 8,778935 | 7,18306 |

Как видно из результатов таблицы, целесообразно исследовать показатели на больших количествах прямоугольников. Ниже будут представлены графики зависимости времени работы алгоритма от количества узлов.

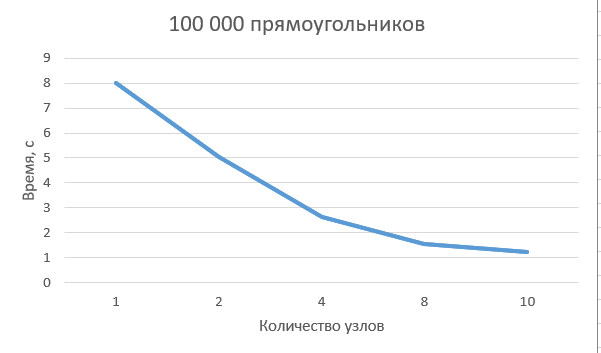


Рисунок 1. Зависимость времени работы алгоритма от количества узлов при 100 000 прямоугольников.

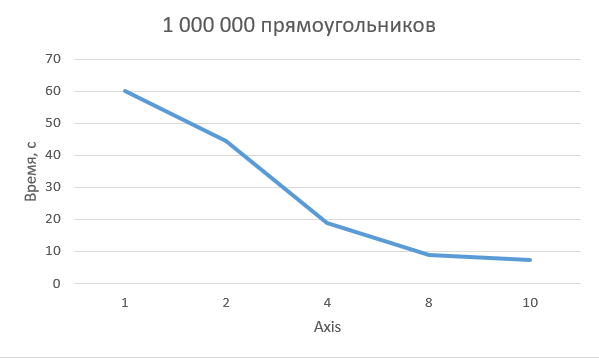


Рисунок 2. Зависимость времени работы алгоритма от количества узлов при 1 000 000 прямоугольников.

В таблице ниже представлены расчёты ускорения и эффективности для 1 000 000 прямоугольников.

|  |  |  |
| --- | --- | --- |
| Кол-во узлов | Ускорение | Эффективность |
| 1 | 1 | 1 |
| 2 | 1,346667 | 0,673333 |
| 4 | 3,166626 | 0,791656 |
| 8 | 6,838208 | 0,854776 |
| 10 | 8,357466 | 0,835747 |

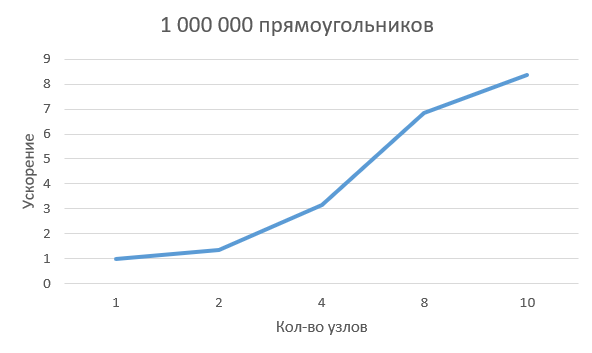


Рисунок 3. Зависимость ускорения от количества узлов



Рисунок 4. Зависимость эффективности от количества узлов

Также алгоритм был дополнительно ускорен при помощи OpenMP. Результаты замера времени (в секундах) выполнения алгоритма представлены в таблице ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Кол-во узлов / кол-во прямоугольников | 1 | 2 | 4 | 8 | 10 |
| 100 000 | 2,274521 | 2,15474 | 1,499321 | 1,239097 | 1,259955 |
| 1 000 000 | 15,109148 | 11,953504 | 5,284758 | 2,930465 | 2,849221 |

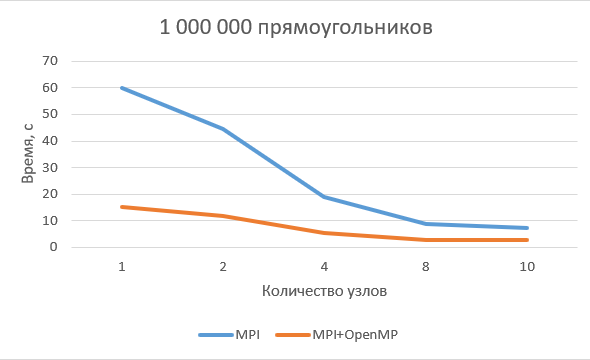


Рисунок 5. Зависимость времени выполнения алгоритма от количества узлов для двух вариантов алгоритма.

Ниже представлены вычисления ускорения и эффективности реализации алгоритма с применением OpenMP в сравнении с вариантом реализации без него.

|  |  |  |
| --- | --- | --- |
| Кол-во узлов | Ускорение | Эффективность |
| 1 | 3,973234 | 3,973234 |
| 2 | 5,022141 | 2,511071 |
| 4 | 11,3595 | 2,839874 |
| 8 | 20,48555 | 2,560694 |
| 10 | 21,06968 | 2,106968 |

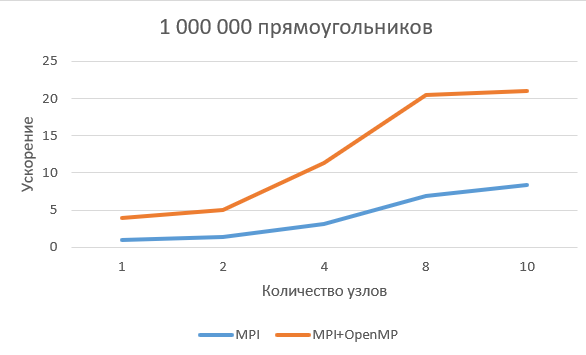


Рисунок 6. Зависимость ускорения от количества узлов для двух вариантов реализации алгоритма.

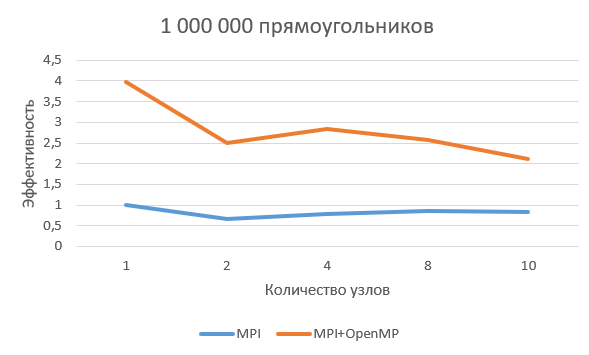


Рисунок 7. Зависимость эффективности от количества узлов для двух вариантов реализации алгоритма.