МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Нижегородский государственный университет им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Институт информационных технологий, математики и механики**

**ЛАБОРАТОРНАЯ РАБОТА**

**Построение выпуклой оболочки – проход Грэхема**

**Выполнил:** студент группы 381506-1

Грибов Михаил Николаевич

Нижний Новгород

2018

**Содержание**

[**Постановка задачи** 3](#_Toc514715059)

[**Метод решения** 4](#_Toc514715060)

[**Схема распараллеливания** 5](#_Toc514715061)

[**Описание программной реализации** 6](#_Toc514715062)

[**Generator** 7](#_Toc514715063)

[**Checker** 8](#_Toc514715064)

[**Результаты экспериментов** 9](#_Toc514715065)

# **Постановка задачи**

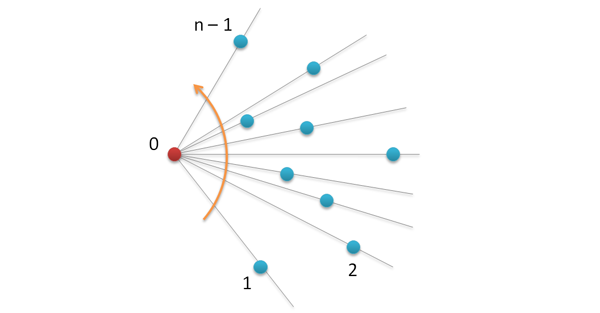
В двумерном пространстве задано множество точек {(x1,y1), (x2,y2), (x3,y3), … , (xn,yn)}. Необходимо построить выпуклую оболочку использую проход Грэхема.

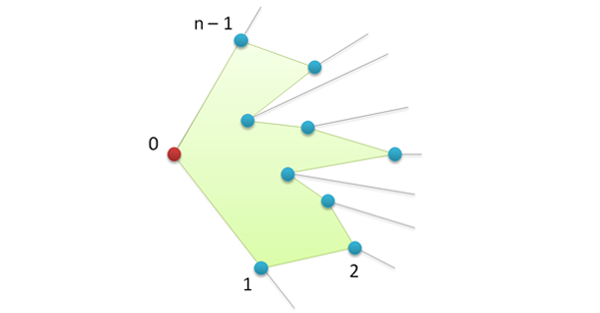
Решение должно содержать:

1. Последовательное решение задачи
2. OpenMP версию
3. TBB версию

# **Метод решения**

Сначала выбирается точка, которая точно будет включена в выпуклую оболочку. В моем решении выбрана точка с минимальной абсциссой. Далее все точки сортируются по возрастанию полярного угла начиная с выбранной точки. Первые две вершины включены в оболочку. Просматриваем все остальные вершины, и отслеживаем направление поворота в них с точки зрения двух последних точек. Если это направление отрицательно, то можно срезать угол удалением последней вершины. Как только поворот оказывается положительным, срезание углов завершается, текущая вершина добавляется к оболочке.





# **Схема распараллеливания**

1. OpenMP

Исходное множество делится на n множеств по количеству потоков. Каждый поток строит выпуклую оболочку для своей части множества. Главный поток в дополнение обрабатывает остаток от деления множества на число потоков. Затем главный поток производит слияние полученных оболочек.

1. TBB

Задается итерационное пространство с диапазоном от нуля до размера рассматриваемого множества, параметр grainsize равен отношению размера множества к числу заданных потоков. Полученные оболочки редуцируются и происходит их общее слияние.

# **Описание программной реализации**

В заголовочном файле Sol.h описаны:

Класс CPoint, вычисление определителя для двух точек и основные операции.

Метод QuickSort – быстрая сортировка по величине полярного угла.

Метод Conv – вычисление выпуклой оболочки.

В файле Before\_code.cpp происходит вычисление результата и запись его в бинарный файл.

# **Generator**

Генератору в аргументы командной строки передаются “название\_создаваемого файла” и “число”(номер теста от 0 до 19). Количество точек множества задается по номеру теста из массива

{ 10, 100, 200, 400, 1000, 2000, 4000, 10000, 20000, 50000, 100000, 200000, 400000, 800000, 1000000, 2000000, 4000000, 8000000, 10000000, 20000000 }.

Для каждой точки случайно генерируются две координаты: X и Y. Эти координаты последовательно записываются в бинарный файл.

Последние четыре точки описывают граничные условия и задаются отдельно: (-1000; 1000), (1000; 1000), (1000; -1000), (-1000;-1000).

Эти значения позволяют далее проверить корректность результатов.

# **Checker**

Для проверки корректности результата вычисленное значение сравнивается с эталонным. Эталонное значение задается точками (-1000; 1000), (1000; 1000), (1000; -1000), (-1000;-1000) и считывается из файла. Решение считается верным если набор точек выпуклой оболочки целиком совпадает с заданным набором.

# **Результаты экспериментов**

Сравнение последовательной и openmp версий(время в секундах)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кол-во точек | Послед. версия | Openmp  2 потока | Openmp  4 потока | Openmp  8 потоков |
| 100000 | 0.0112753 | 0.00601694 | 0.00299857 | 0.00312475 |
| 200000 | 0.0246665 | 0.0120076 | 0.00606951 | 0.00593066 |
| 400000 | 0.0484188 | 0.025473 | 0.0125392 | 0.013253 |
| 800000 | 0.101537 | 0.0529618 | 0.0346064 | 0.027417 |
| 1000000 | 0.130379 | 0.0667835 | 0.0351705 | 0.0341094 |
| 2000000 | 0.2695 | 0.139589 | 0.0726508 | 0.0708735 |
| 4000000 | 0.664005 | 0.295172 | 0.160709 | 0.144608 |
| 8000000 | 1.16718 | 0.607892 | 0.320413 | 0.291802 |
| 10000000 | 1.43678 | 0.883832 | 0.407082 | 0.354405 |
| 100000000 | 14.6091 | 8.30444 | 4.22047 | 4.24457 |