Harsimrat Singh
2015CS50284

COL380
Assignment 1

Problem 2

In this, I have introduced a new vector of pair which is of int, names hull, which stores the final output hull. Also, I have defined 3 new functions aside from given function calcConvexHull() - lineSide(), distFromLine(), hullRecFunc(). Details of all these are given below:

1.) **calcConvexHull() -** In this function, given its declaration, I first set number of threads that I am going to use which is argument to this function num_threads. Then, I set nested flag to 1 so I can use nesting of fork. Since to compute convex hull we need atleast 3 points, otherwise it should return zero sized vector. Now I assumed coordinates as index of given image. Then I found those points which has maximum and minimum value of x. As these points are going to be part of convex hull so I push them into our final output vector hull. Now we have to perform convex hull algorithm, so we have to find point which is maximum distance apart from line formed by these points on either side of this line. So we call hullRecFunc() 2 times which does so. For parallelizing code we will use sections and calling of hullRecFunc() function will be done in section. Finally, before returning our final vector hull we have to be sure there is no duplicates. So I converted this vector into set and then back to vector then returns it

2.) **lineSide()** - This function takes 3 arguments - 3 pairs of int. First and second pair forms a line and we have to tell whether third pair falls on either side of line or on the line. It returns integer value 0, 1, -1. 0 if it lies on line and 1, -1 if on either side.

3.) **distFromLine()** - This function again takes 3 arguments - 3 pairs of int. First and second pair forms a line and we have to tell perpendicular distance of third pair from the line. Hence it returns integer value i.e. value of distance.

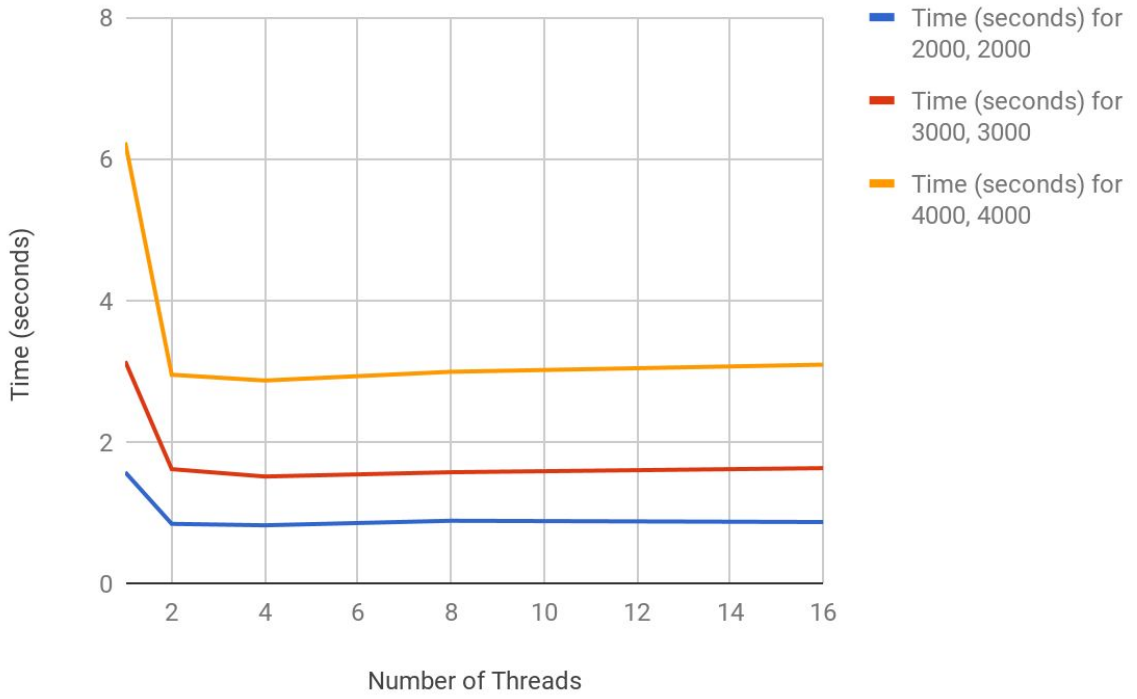
4.) **hullRecFunc()** - This function is called by calcConvexHull() which provides it 4 arguments - one is image, coordinates of 2 points and one more variable side.What this function does is, it first finds a point from image which is maximum distance apart from line formed by line given by 2 points. I have implemented it parallely. I gave each thread a row and then computed maximum distance for points in a column and then sequentially computed maximum distance out of these maximum distances. I also stored coordinates of this point. If we are not able to find any point then it means I have to return void from this function, else we push back this new point in our final output vector hull. And then we join new point with given 2 points and then again do this i.e recursively call

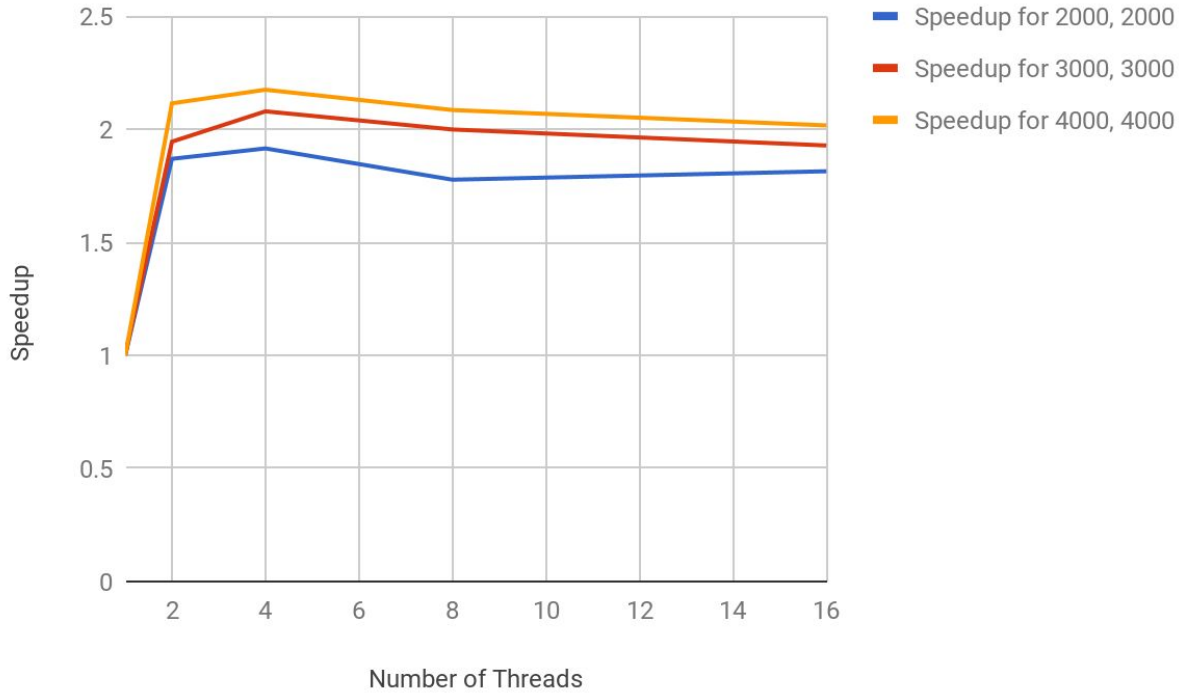
hullRecFunc() for new point with each of given points. I again used sections in implementing this.

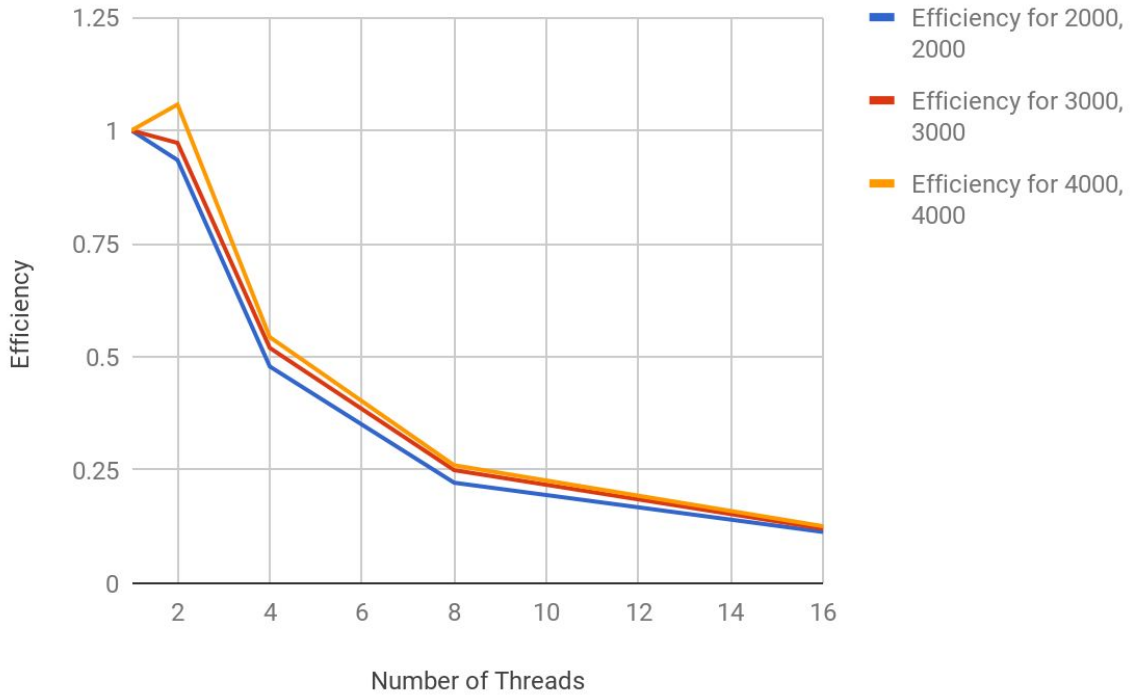
These are observation I get after running my code on input of height of 10000 and width of 1000

Number of Threads	Time (seconds) for 2000, 2000
1	1.57
2	0.84
4	0.81971
8	0.88365
16	0.865454

We can see that increase number of threads time taken by code decreases.


Time (seconds)
8
6
4
2
0
2
4
6
8
10
12
14
16
Number of Threads
Time (seconds) for 2000, 2000
Time (seconds) for 3000, 3000
Time (seconds) for 4000, 4000


Speedup
2.5
2
1.5
1
0.5
0
2
4
6
8
10
12
14
16
Number of Threads
Speedup for 2000, 2000
Speedup for 3000, 3000
Speedup for 4000, 4000


1.25
1
0.75
0.5
0.25
0
Efficiency
2
4
6
8
10
12
14
16
Number of Threads
Efficiency for 2000, 2000
Efficiency for 3000, 3000
Efficiency for 4000, 4000