Chris Ruiz

Meghan Haukaas

Stephen Belden

2015-Oct-19

**Finding Collinear Points Tests & Discussion**

**Test Cases**

1. Zero points in the list  
   Points:  
   Expected output: Not enough collinear points.  
   Actual output:  
   Not enough input points to find collinear points.
2. No lines of more than 2 collinear points.  
   Points: (0,0) (1,1) (1,2) (2,3) (0,2) (10,0) (5,3) (8,2) (2,8) (6,7)  
   Expected output: Not enough collinear points.  
   Actual output:  
   No groups of 4 or more collinear points.
3. No lines of more than 3 collinear points.  
   Points: (0,0) (1,1) (2,2) (2,3) (0,2) (10,5) (5,3) (2,4) (3,8) (5,2)  
   Expected output: Not enough collinear points.  
   Actual output:  
   No groups of 4 or more collinear points.
4. Lines of 4 or more collinear points.  
   Points: (0,0) (1,1) (2,2) (3,3) (4,4) (10,5) (6,3) (4,2) (8,4) (3,9)  
   Expected output: Line A: Points at (0,0) (1,1) (2,2) (3,3) (4,4) are collinear.  
   Line B: Points at (10,5) (6,3) (4,2) (8,4) (0,0) are collinear.  
   Actual output:  
   Line 1 has 5 collinear points: (0, 0), (6, 3),  
    (10, 5), (4, 2), (8, 4),

Line 2 has 5 collinear points: (0, 0), (2, 2),

(3, 3), (4, 4), (1, 1),

1. Two parallel collinear lines  
   Points: (0,1) (1,2) (2,3) (3,4) (4,5) (3,2) (4,3) (5,4) (6,5) (7,6)  
   Expected output: Line A: Points at (0,1) (1,2) (2,3) (3,4) (4,5) are collinear.  
   Line B: Points at (3,2) (4,3) (5,4) (6,5) (7,6) are collinear.  
   Actual output:  
   Line 1 has 5 collinear points: (0, 1), (1, 2),

(2, 3), (3, 4), (4, 5),

Line 2 has 5 collinear points: (3, 2), (4, 3),

(5, 4), (6, 5), (7, 6),

1. Vertical and Horizontal lines  
   Points: (0,0) (0,1) (0,2) (0,3) (0,4) (1,0) (2,0) (3,0) (4,0)  
   Expected output: Line A: Points at (0,0) (0,1) (0,2) (0,3) (0,4) are collinear.  
   Line B: Points at (0,0) (1,0) (2,0) (3,0) (4,0) are collinear  
   Actual output:  
   Line 1 has 5 collinear points: (0, 0), (3, 0),

(4, 0), (1, 0), (2, 0),

Line 2 has 5 collinear points: (0, 0), (0, 3),

(0, 2), (0, 4), (0, 1),

**Time Complexity**

Including only the steps after points are loaded/generated:

* Creating edges between every point is θ(N2)
* Sorting all edges is θ(K log K) where K is N2 elements
* Finding collinear points in the sorted list is θ(K), where K is N2 elements

These steps are performed once each, which means that the overall time complexity for the algorithm is equal to the time complexity of the slowest step (that is, the step with the fastest growing number of operations). The slowest step is sorting, which is θ(N2 log N2), or θ(N2 log N), which makes the time complexity of this algorithm θ(N2 log N).

In theory the above is true, but in practice the time complexity only looks like θ(N2 log N) for very large values of N. For small values of N, the N2 steps (especially the last step, which performs a large number of operations on each element) affect the overall completion time significantly. Only when N is very large does the greater time complexity of θ(N2 log N) start to overshadow the effects of the N2 steps.

**Discussion**

All of our tests worked as expected. Actual time complexity matches the goal time complexity.

A minor consideration of the problem involved accounting for double calculation errors within a given precision. C++ doubles, while exceedingly accurate, do face some loss of precision. A set precision, 10 × 10-12, was assigned for the equivalency comparison operation of two Edge objects. Edges are considered equivalent if their slope and intercept values are both within one precision level of each other.