Group 12

Aaron Lemmon

1st Place External Submission

Sorted Correctly at Competition

Presented by Daniel Frazier

Overview for Algorithm

For each point

Compute Squared Euclidean Distances to both reference points.

Store shortest distance stored in array of longs (sortMe[i]) (Signed 64 bit, Max value 2^{63}-1)

i	toSort[i][0]	toSort[i][1]	toSort[i][2]
0	2532826	2532940	0
1	2532784	2532955	1
...	...	...	...

→

i	sortMe
0	0000000419389 2369298
1	0000000419407 3881385
...	...

Overview for Algorithm

For each point

- Compute Squared Euclidean Distances to both reference points.
- Shortest distance stored in array of longs (sortMe[i]) (Signed 64 bit, Max value 2^{63}-1)
- Perform a left arithmetic shift of 20 bits on each sortMe[i] and add i

This frees the last 20 bits,
values from 0 to 1,048,576.

Which allows every index from
i=0 to i=1,000,000-1 to fit within
those last 20 bits.

i	sortMe
0	00000011 11010000 01110111 01110111 00000011 10010010
1	00000011 11010000 10000010 01001000 10101011 00101001
.	...

i	sortMe
0	00111101 00000111 01110111 01110000 00111001 00100000 00000000 00000000
1	00111101 00001000 00100100 10001010 10110010 10010000 00000000 00000001
.	...

Efficiency of Algorithm

It loops once to find the distances, $O(n)$

Then Arrays.sort(long[] arr) which is dual pivot quick sort $O(n^2)$

Then loop through to map indexes $O(n)$

In practice the common case for dual pivot quick sort is $\Theta(n \lg(n))$

Big theta of the common case for this algorithm is $\Theta(2n + n \lg(n)) = \Theta(n \lg(n))$

This algorithm uses 2n dynamic memory for the long array and toReturn array.

Other interesting information

This program breaks when

N > 1,048,576

And there is a very rare chance of breaking when

gridSize > Sqrt(2^(63-20)) = 2965820

These numbers are to the specification within the sorting competition rules.