Code Jam 2020 - Round 2

Analysis: Wormhole in One

Test Set 1

As the limits for Test Set 1 are very small, we should be able to apply a brute-force algorithm and check all possible directions, starting points, and links between holes. However, there are infinitely many directions and starting points. Let's find a way to work with a finite number of possibilities.

First, let us observe that in order for the ball to touch more than two holes, some pairs of holes must be on lines parallel to the chosen direction. Otherwise, the best we can do is to link together two holes; the ball will go through them and will not touch any other holes.

So, when choosing an initial hit direction, we only need to consider those that are parallel to lines that connect pairs of holes.

Also, the exact starting point is not important. We can decide which hole we want to enter first, and then, regardless of which hit direction we choose, we can position the ball such that it will enter that hole (before any others). For example, because the holes are always at unique integer coordinates, we can choose a starting distance of 0.1 from the hole, in the direction that is the opposite of our hit direction.

Now, we can try all possible starting holes and linking schemes — with a recursive backtracking algorithm, for example — and choose the combination that touches the largest number of holes. This should be enough to pass Test Set 1.

Test Set 2

Suppose for now that we have chosen the direction of the hit. Let's calculate the maximum possible answer given that decision.

Imagine lines parallel to the chosen direction and going through all of the holes. Notice that each hole is on at most one such line. We will call a line an *odd line* if it contains an odd number of holes (but more than one), and an *even line* if it contains an even number of holes (at least two). If there are lines which only contain one hole, we call these holes *standalone holes*.

Let's also consider the holes along each line to be ordered in the chosen direction.

Note that:

- We cannot touch more than two standalone holes: one at the beginning, and one at the very end of the ball's journey.
- In the best case, we would touch all non-standalone holes.

Let's say we have C_{odd} total holes on the odd lines, C_{even} total holes on the even lines, and C_1 standalone holes. Then the answer is not greater than $C_{odd} + C_{even} + \min(2, C_1)$.

To touch two standalone holes, we should touch an even number of holes between them. To understand why, note that first of the standalone holes must be an entry to a wormhole, and the second one must be an exit from another wormhole. All of the holes that the ball touches in between those starting and ending holes must be linked in pairs by wormholes, which means there should be an even number of holes. This also means that if the number of non-standalone holes is odd, then we will not be able to touch two standalone holes; in such a case, the answer will not be greater than C_{odd} + C_{even} + min(1, C_1).

As we will see, these upper limits can actually be achieved, so the answer is:

- C_{odd} + C_{even} + min(1, C_1), if C_{odd} + C_{even} is odd
- C_{odd} + C_{even} + min(2, C_1), if C_{odd} + C_{even} is even

Note that the parity of C_{odd} + C_{even} is the same as the parity of C_{odd}, as C_{even} is always even.

Let's construct the linking scheme for the case when C_{odd} is even and C_1 is greater than 1:

1. Connect one standalone hole to the first hole of any even line.
2. Connect the other holes on that line pairwise consecutively (the last hole will remain unconnected).
3. Connect the last hole on that line to the first hole of another even line, and repeat steps 2 and 3 until only odd lines are left untouched.
4. Connect the last hole of the last even line to the first hole of any odd line.
5. Connect the second hole of that odd line (A) to the second hole of another odd line (B).
6. Connect the last hole of line B to the first hole of line B.
7. Connect the last hole of line A to the first hole of another odd line.
8. Connect the remaining holes of line A in consecutive pairs.
9. Connect the remaining holes of line B in consecutive pairs.
10. Repeat steps 5-9 until all the odd lines are used. When there are no more odd lines, connect the last hole of the last odd line to an unused standalone hole.

This scheme can be easily modified for the other cases, when C_{odd} is odd, and/or C_1 is less than 2.

Summarizing this, we can make full use of all of the odd and even lines and up to two standalone holes.

To calculate the number of holes on each line for a given direction, we can iterate through all ordered pairs of the holes and find the equation of the line that connects them in the form $y = mx + y_0$. Now, for each m, we store how many times we see each y_0. This number will be equal to the number of pairs of holes on this line, which we can use to calculate the number of holes. Note that we only need to do this once, as we will get the counts for all directions we are interested in.

Now, we can iterate through all the directions and calculate the answer for each one as described earlier by iterating through all the lines parallel to the current direction. Our final answer will be the maximum over the answers for all directions.

Note that even though there are $O(\mathbf{N}^2)$ directions, and $O(\mathbf{N})$ lines parallel to a direction, the total number of lines for all directions is $O(\mathbf{N}^2)$, as every line can be parallel to two (opposite) directions. So the total time complexity of this solution implemented optimally is $O(\mathbf{N}^2)$, but slower implementations might also pass.