Code Jam 2008 - APAC Semifinal

Analysis: What are Birds?

The Simple Solution

Let us visualize this problem. Each animal is characterized as a pair (H, W), which can be viewed as a point in the two dimensional Cartesian coordinate system. For each animal that is known to be a bird, we color it red. For each animal known to be a non-bird, we color it blue. The problem states that there exists a rectangle such that a point is red if and only if it is in the rectangle. The problem is trivial if there are no red points. From now on we assume there are red points.

For any two distinct points U and V, there is naturally a rectangle determined by U and V. It is the smallest rectangle containing both U and V, with the four corners U, V, (H_U, W_V), and (H_V, W_U). When U and V are on the same horizontal or vertical line, the rectangle degenerates to a segment. Formally, the rectangle contains all the points (H, W) such that

$$(|H - H_U| + |W - W_U|) + (|H - H_V| + |W - W_V|) = |H_U - H_V| + |W_U - W_V|.$$

We have the following proposition.

> If U and V are two red points, and R be the rectangle determined by U and V; then any point in R must also be red, since any rectangle containing both U and V must contain R.

For the set of all red points in the input, there is also naturally a *smallest* rectangle, R_0, that contains all of them. There are different ways to define this rectangle:

- (a) The intersection of all the rectangles that contain all the red points.
- (b) The union of all the rectangles determined by U and V, where (U, V) range over all the red point pairs.
- (c) The rectangle with four corners A=(H_{min}, W_{min}), B=(H_{min}, W_{max}), C=(H_{max}, W_{min}), and D=(H_{max}, W_{max}), where the min and max are taken over all the red points.

The interested reader may check the equivalence of the above definitions.

Now, given a point X, we want to know if it is a bird or not. Clearly, if X is inside R_0, then it must be bird. Otherwise, we may pretend it is red, and compute the new rectangle R' based on (c). This can be done in a constant number of comparisons. If there is a blue point from the input that lies in R', then X must not be a bird. Otherwise, X might (taking R' as the red rectangle) or might not (taking R_0 as the red rectangle) be a bird.

By the limits of this problem, a Θ(NM) algorithm is fast enough. That is, we simply take any blue input point, and check if that is inside R'.

Further Discussion

Another way to view the last step of the solution, when the query point X is given, is to see whether there is a known red point Y such that the rectangle determined by X and Y contains any blue point. It is enough to check Y = A, B, C, or D as in (c). To check this, we do not need to go over all the blue input points. There are standard pre-processing techniques that can reduce the query time from N to log N.

For example, for A, we can define two sets based on the blue inputs: those points that are higher (along the W-axis) than A (call them A_1) and the rest of the A points ($A_2 = A - A_1$). The task of determining whether there is a blue point between X and A becomes the query of lowest or highest point in the range between H_X and H_A.