CS 161 (Stanford, Fall 2025) Section 7 Extra

1 Pareto Optimal

Given a set of 2d points P, a Pareto optimal point is a point (x, y) such that $\forall (x', y')$ we have either $x > x'$ or $y > y'$. Develop an algorithm to to find all Pareto optimal points.

Solution

First sort the points by decreasing x coordinate and break ties using y. Clearly in this sorted list the first point is guaranteed to be Pareto optimal (as it has the largest x coordinate and the largest y amongst points with that x).

Now, as we progress down our input list, each subsequent point has either equal or smaller x coordinates. Therefore, the only way that it could possibly be a Pareto point is if it has a larger y coordinate than the previous points we have seen. This is because this point has a larger y coordinate than the previous points we have seen (since it broke the Y record), a larger y coordinate than the points with the same x coordinate (since we break ties by y), and a larger x coordinate than the rest of the points (since the list is sorted by x).

Therefore, we save the value of the largest y coordinate we have yet encountered, and a Pareto point lower in the list must have a larger y coordinate than our saved value.

The total runtime is $O(n \log n)$ due to sorting.

```
Data: list of points (x_n, y_n)
sort input points (largest first) by x coordinate, then tiebreak by y
initialize pareto points array with first point from the sorted list
set Y to y coordinate of that point
for point in sorted input do
    if y coordinate of point > Y then
        add point to pareto points array
        update Y to the y coordinate of this point
return pareto points array
```

2 Roads and Airports

Given a set of n cities, we would like to build a transportation system such that there is some path from any city i to any other city j. There are two ways to travel: by driving or by flying. Initially all of the cities are disconnected. It costs r_{ij} to build a road between city i and city j. It costs a_i to build an airport in city i. For any two cities i and j, we can fly directly from

i to j if there is an airport in both cities.

Give an efficient algorithm for determining which roads and airports to build to minimize the cost of connecting the cities.

Solution

To find the roads and airports to build, we first note that there are two cases: either we do not build any airports or we build at least one airport.

To consider the case where we do not build any airports, we construct an undirected graph where the cities are the nodes and the roads are the edges with weights corresponding to the cost of building that road. We then construct the MST of this graph. This gives us the minimum construction cost using no airports. (If we constructed a non-tree connected graph, we could always remove a road to decrease cost without disconnecting the graph, so the optimal solution must be a tree.)

Then we consider the case where we choose to build at least one airport. To model this, we construct a slightly different graph. We start with the same graph from the previous case: an undirected graph where the cities are the nodes and the roads are the edges with weights corresponding to the cost of building that road. We then add another node to the graph, representing the air. Call this node a. We add an undirected edge between every city i and a with weight a_i. We then construct the MST of this graph.

To find the overall minimum cost set of roads and airports to build, we use either the MST from the first case or the MST from the second case, whichever has lower total cost. For every edge in the MST between two cities i and j, we build road between i and j, or for every edge between a city i and a the airport node, we build an airport in city i.

The total runtime is $O(n \log n + m)$ by using a Fibonacci Heap to implement the Prim's algorithm.