The Bitonic Euclidean Traveling Salesman Problem

Assume that we have n points (x_i, y_i). Sort them by increasing x coordinates, so that $x_{i-1} < x_i, 1 \leq i < n$. Call p_i the resulting set of points, which are strictly going left to right.

Consider the shortest bitonic tour on the first i points $p_1, ..., p_i$. Take the edge (p_k, p_i) with $k < i-1$, and observe that the shortest bitonic tour on the points $p_1, ..., p_i$ is the shortest bitonic tour on $p_1, ..., p_{k+1}$ minus the edge (p_k, p_{k+1}) plus the edge (p_k, p_i) and plus the path $\{(p_{k+1}, p_{k+2}), (p_{k+2}, p_{k+3}), ..., (p_{i-1}, p_i))\}$.

So for the length of the shortest bitonic tour on the first i points we can write the following recursion:

$$l(i) = min_{1 \leq k < i-1}\{l(k+1) + D(k+1, i) + d(k, i) - d(k, k+1)\}, \quad 3 \leq i \leq n$$

$$l(2) = 2d(1, 2)$$

where:

$$d(i, j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

$$D(i, j) = \sum_{k=i}^{j-1} d(k, k+1)$$

The algorithm finds the shortest path while computing $l(i)$, for $i = 1, 2, ..., n$. It starts with $P_3 = \{(p_1, p_2), (p_2, p_3), (p_3, p_1)\}$ and adds and deletes edges from P_i in the same way that it adds and deletes distances on $l(i)$. More precisely, for k that minimize $l(i)$ it deletes from P the edge (p_k, p_{k+1}) and it adds the edges (p_k, p_i) and $\{(p_{k+1}, p_{k+2}), (p_{k+2}, p_{k+3}), ..., (p_{i-1}, p_i))\}$.

We can make the running time $O(n^2)$ if we precompute and store in a table $D(i, j), 1 \leq i < j \leq n$. Doing this requires $O(n^2)$ time, filling the table left to right.

Then the evaluation of the expression within brackets for $l(i)$ takes $O(1)$, and since it is repeated $O(n)$ times, at each i the cost of computing the $l(i)$ is $O(n)$. The cost updating the path P_i is $O(n)$ as well. So the total cost is $O(n^2)$.