**Dynamic Programming for Minimum Steiner Trees**

***Abstract***

We are here presenting a dynamic programming algorithm which solves the Minimum Steiner Tree Problems with k terminals with O\*(ck) time complexity for all c > 2. This improves the running time of the earlier fastest exponential time algorithms, which was Dreyfus-Wagner algorithm of order O\*(3k) and the so-called as “full set dynamic programming” algorithm, solving rectilinear instances in time O\*(2.38k).

***Introduction***

The Steiner tree problem is one of the most popular NP-hard problems. Given a graph G = (V, E) of order n = |V|, edge costs c: E → R+ and a set Y ⊆ V of k = |Y| terminals, we are to find a minimum cost tree T ⊆ E connecting all terminals. Note that, we identify a subtree of the underlying graph with its edge set T ⊆ E. The node set of the tree is denoted by V (T). So an Optimal Steiner Tree for Y is a tree T = T(Y) that minimizes the value of c (T) subject to Y ⊆ V (T).

***Algorithm ASC (“Attach Small Components”)***

1. For each Y˜, Y ⊆ Y˜ ⊆ V, |Y˜| = k + [1/e] do:
2. Compute T(X) for all X ⊆ Y ˜, |X| ≤ €k + 1.

For all X ⊆ Y˜, |X| > €k + 1, compute T(X) recursively, according to

T(X) = min {T(X1) ∪ T(X2)|X = X1 ✶ X2, |X2| ≤ €k + 1}.

There are O (n1/e) choices for Y˜ of size k˜ = k + [1/e]. The time needed for 1) (using Dreyfus-Wagner) is negligible for reasonably small € > 0. So the total running time is bounded by

n1/e Ei (k˜/i!) (i/€k + 1) ≤ n1/e k˜ 2k˜ ((k˜/2)/ €k˜). This yields our main result.