**The Dynamic Steiner Tree Algorithm**

By: Thomas Wheeler, Paul Nathan, and Weir King

**Introduction**

This report analyzes the modifications made to the current Steiner tree by the dynamic Steiner tree algorithm for different vertex removals. The changes to the tree and the subsequent altering done by the Steiner tree algorithm are graphically displayed.

*What is a Steiner tree?*

Let graph G consist of vertices V, edges E and additionally terminals T ⊂V.

A Steiner tree for G is a subgraph of G with minimum edge weight that contains all T.

The growth of the internet has emphasized the importance of scalable and flexible multipoint communication networks. The challenge is to optimally connect a set of routing nodes and clients that changes dynamically with time. However, since arbitrarily reconnecting networks could severely interrupt lengthy transitions, we would like to do so in a manner that is conservative in the sense that it gives preference to preserving the existing connections as taking network optimality into account.

Mathematically, this problem can be represented as a dynamic steiner tree.

*What is the dynamic Steiner tree problem?*

1. Let graph G be as above and S be the Steiner tree for G.
2. A modification step is performed on G:
   1. Add or remove a vertex to V

**OR**

* 1. Add or remove a terminal from T

1. Compute the new Steiner tree S` for the new modified graph G`

The same real world example of a communication network can again be used to represent the dynamic Steiner tree problem. In an already established communications network endpoints are occasionally added or removed; routers can fail and new routers can be added. A dynamic Steiner tree algorithm can be used to find the new Steiner tree for the new communications network.

**Hypothesis**

The alterations made to the Steiner tree by the Steiner tree algorithm will not be visually significant unless certain key vertices are added or removed by modifications.

**Related Work:**

The dynamic Steiner tree problem was first proposed in 1991 by Imase and Waxman. They designed the dynamic algorithm we have implemented and an additional algorithm that does not re-route the existing Steiner tree for each modification. Imaxe and Waxman however do not visualize the real-time changes made to the Steiner tree after each step. Online implementations, as well as implementations that take Euclidean plane have been developed previously by Alon, Azar et al. Our work focuses solely on the dynamic steiner tree problem in it classical formulation and does not take these Euclidean assumptions into accound.

**Approach:**

We verified the approach of Imase and Waxman, by implementing the **Edge Bound Algorithm** for reconfigurable dynamic steiner tree networks in a portable python 3 package.



The Edge Bound Algorithm, adapted from Imase and Waxman (1991)

Their formulation of the problem begins with an initial node in the tree. Changes are then made in the form of change requests, with which EBA runs an **add** or **remove** routine.

**Experimental Design:**

Our implementation generates random points in a 2-Dimensional plane, for easy visualization. We then encode these points into an weighted adjacency matrix, and generate a sequence of requests to either add or remove a node. We then use pythons matplotlib library to visualize the graphs changing over time. We gain an intuitive understanding of the algorithm by simulating and visualizing several graphs with varying values of the *delta. Delta* in the context of the Extended Bound algorithm is conceptually a preference for preserving the existing graph connections, unless a substantially better configuration is available. We gain insight to the runtime of the algorithm by using our testing framework to experimenting with it on graphs different size.

**Conclusions and Future Work:**

The distributed nature of the internet as well a distributed algorithms for the steiner tree problem, which are inherently scalable and could be a promising avenue for future work. Also, because the general problem is NP hard (by reduction to 3SAT) approximation algorithms under problem specific assumptions are and will continue to be an fruitful avenue for research.

As we adventure into the future, we stand to see the expansion of the internet to what is project to be as much as 5 billion people as soon as 2020 (by some predications) as well as the rise of the internet of things, and multi-robotic networked systems. Network demands will become heavier, and more dynamic than ever. Our civilization will see some of it’s the most wonderful implications of these technologies as well as the great challenges that come with them in the coming decades. This trajectory of growth implies that scalable and flexible dynamic multipoint communication is and will increasingly become a fundamentally important theoretical and practical problem in computer science.