Borah’s 1-Steiner Algorithm

James Yang
### Algorithm Refresher

- **Find EVERY Node / Edge Pair**
  - For each pair, find the longest edge from the node to the edge
  - Compute gain for each pair
    - gain = length of longest edge – length from steiner point to node

- **Sort the pairs in descending order**

- **For each pair,**
  - Add steiner point (p)
  - Remove longest edge (e2)
  - Remove the edge (e1)
  - Connect all 3 nodes to the steiner point
Demo

1. Dr. Lim’s Example
2. 20 Random Points
3. 100 Random Points
4. 200 Random Points
5. 500 Random Points
Dr. Lim’s Example

- Initial
- Final

Gain = 3
Demo (20, 100, 200 and 500 Nodes)

- Journal’s Results

<table>
<thead>
<tr>
<th>Size</th>
<th>Passes Required</th>
<th>Avg</th>
<th>Max</th>
<th>Min</th>
<th>Avg. improv./pass (%)</th>
<th>Max. improv./pass (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Avg</td>
<td>Max</td>
<td>Min</td>
<td>1st</td>
<td>2nd</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>1.41</td>
<td>3</td>
<td>1</td>
<td>8.88</td>
<td>.99</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>1.7</td>
<td>3</td>
<td>1</td>
<td>9.15</td>
<td>.97</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>2.3</td>
<td>4</td>
<td>1</td>
<td>9.08</td>
<td>1.31</td>
</tr>
<tr>
<td>100</td>
<td></td>
<td>2.5</td>
<td>5</td>
<td>2</td>
<td>9.28</td>
<td>1.40</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>2.7</td>
<td>4</td>
<td>2</td>
<td>9.48</td>
<td>1.36</td>
</tr>
<tr>
<td>500</td>
<td></td>
<td>3.3</td>
<td>4</td>
<td>3</td>
<td>9.48</td>
<td>1.41</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Size of net</th>
<th>Batched 1-Steiner</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>7.1 msec</td>
</tr>
<tr>
<td>6</td>
<td>13.3 msec</td>
</tr>
<tr>
<td>8</td>
<td>32.1 msec</td>
</tr>
<tr>
<td>10</td>
<td>59.6 msec</td>
</tr>
<tr>
<td>20</td>
<td>456 msec</td>
</tr>
<tr>
<td>50</td>
<td>6.53 sec</td>
</tr>
<tr>
<td>100</td>
<td>52 sec</td>
</tr>
<tr>
<td>200</td>
<td>395 sec</td>
</tr>
<tr>
<td>500</td>
<td>&gt;1.5 hrs</td>
</tr>
<tr>
<td>1000</td>
<td>-</td>
</tr>
</tbody>
</table>

With 1994 Technology
Probably a Pentium I
Uh oh.

- **Missing Nodes**

- **H. Zhou’s Claim**
  
  From “Efficient Steiner Tree Construction Based on Spanning Graphs”

  - Borah’s Algorithm is “not totally correct”
  - ..“we should avoid placing the correctness of an algorithm only on our intuitions”
What can go wrong?

- **Starting Pair**
  - Edge: a to b
  - Node: c

- **Longest Edges**
  - d to e
  - f to g

We pick the closest edge
  - d to e
What can go wrong? (continued)

- **Next Pair**
  - Edge: a’ to b’
  - Node: c’

- **Longest Edges**
  - d to e AND f to g
  - We pick the closest edge for consistency – f to g
My Implementation

- Written in ANSI C and Python (GUI)
- No memory leak (checked using valgrind)
- Uses lots of CPU and Memory
  - CPU at 100% during the execution
  - For >500 nodes, it starts to use page files

Challenges
- Bidirectional Graph
- Lots of book keeping

Possible Improvements
- Use an array or matrix based data structure
- Apply Zhou’s modifications to the algorithm to make it “correct”
End

Thank you