Naïve 1-Steiner by Kahng/Robins

Felix Wu
Implementation

- Prim’s MST Algorithm
- Runs MST Algorithm exhaustively on Steiner points
- Possibilities:
  - Adjacency Matrix
  - Adjacency List
  - Use Heap to keep track of all connecting edges to MST
What is a Fibonacci Heap?

- A heap is a ordered tree
- Fibonacci Heap is a collection of min-heap trees with a pointer to the minimum root node
• Edges are kept as heap nodes in the tree
• We know what endpoints are closest to our current MST
• Amortized Runtime: O(E + V log V)
Test Results: 5 Demand Points
WL = 21
WL = 19

Red point is Steiner point with best gain

Steiner: Pass 1 WL=19, added (3,3)
WL = 18
Test Results: 100 Demand Points
WL: 240
Problem?

WL: 239
Problem?

WL: 238
WL: 219
After 22 Passes (Iterations)
This problem may affect Radius drastically.
This problem may affect Radius drastically
Total Run Time - 2516 ms
First pass - 1683 ms (~67% of run time)
Lowest Wirelength - 219
Slight Modification

- Observation: gain is small per iteration (~1,2)
- Shuffle the Steiner set and break loop immediately after finding positive gain
- No need to compute MST for every Steiner point
- Disadvantage:
  - May lead to inconsistent results
Comparisons

Original (WL = 219, 2516ms)

Modified (WL = 220, 2030ms)
Total Run Time (Avg/5 Runs) - 2030 ms
Ranged from 1764 ms to 2228 ms
Lowest Wirelength (Avg) - 219.4
Improvements

• Make 1-Steiner radius-sensitive