Goal: Connect all nodes in a set while minimizing total wire length and radius.

The tradeoff between wire length and radius represents the ever-present tradeoff between area and performance.

We want a way to construct solutions at varying levels of weight between the two constraints.

Fig. 1. An example where the cost of a shortest path tree (right) is $\Omega(|N|)$ times larger than the cost of a minimum spanning tree (left).
Two Solutions:

Bounded Prim’s Algorithm (BPRIM)
- Connect new nodes to current tree using minimum wire length
- Set a radius bound based on the maximum source-to-node distance
- Compute “look-back” radius bound using variable $\varepsilon$

Bounded Radius Bounded Cost (BRBC)
- Start by constructing a minimum spanning tree using PRIM (i.e. BPRIM with $\varepsilon = \infty$)
- Record the nodes visited on a depth first tour of a rooted tree version of the MST
- Traverse the list of nodes keeping track of visited edge weights
- Add edges as necessary
**Bounded PRIM**

- Start with source node S
- Compute distances from all connected nodes to all disconnected nodes
- Make connection with minimum wire length

Institute boundary on maximum source-to-sink path based on variable $\varepsilon$

Compute “look-back” radius boundary based on maximum source-to-node distance
Bounded PRIM
Specifics of our Implementation

- Take inputs to set grid length, grid width, number of nodes, and epsilon

- Use inputs to generate a random array of nodes in the grid. Set source node as the first in the set

- Keep three matrices: one for connected nodes, one for currently disconnected nodes, and one for the original point set

- Using three nested loops
  - Inner-most: Loop through possible connections to current root node
  - Second: Loop through possible root nodes in connected nodes matrix
  - Outer-most: Loop until all nodes are connected

- If minimum connection breaks “epsilon bound,” trace back towards source node until connection doesn’t break “radius bound.” Ties broken by first-found minimum connection

- Each connected node keeps track of its parent node and the radius up to that node
Let’s start with a small set of nodes at $\varepsilon = 0.5$

Results:
- Grid size: 10 x 10
- Number of nodes: 10
- Total Wire length: 35
- Maximum Radius: 13
- Total Runtime: 0.639 ms
Bounded PRIM
Demo with small point set

How would the graphs differ if we used zero or infinity with this small point set?

\[ \epsilon = 0 \]

**Total Wire length:** 35
**Maximum Radius:** 15
**Total Runtime:** 0.741 ms

\[ \epsilon = \infty \]

**Total Wire length:** 28
**Maximum Radius:** 17
**Total Runtime:** 0.727 ms
Now we perform BPRIM with a medium set of nodes at $\varepsilon = 0.5$

Results:
- Grid size: 50 x 50
- Number of nodes: 50
- Total Wire length: 377
- Maximum Radius: 131
- Total Runtime: 1.056 ms
Lastly, we perform BPRIM with a large set of nodes at $\varepsilon = 0.5$

Results:
- Grid size: 100 x 100
- Number of nodes: 100
- Total Wire length: 900
- Maximum Radius: 235
- Total Runtime: 7.417ms
Start with a MST
Perform a depth first tour of a rooted-tree version of the MST, recording the nodes to a list, L.

Traverse L while computing S (total visited edge weight)
If $S > \varepsilon \times \text{dist(source, current node in L)}$, reset $S$ to 0 and add an edge connecting the current node to the source
After traversing L, compute a shortest path tree of the graph
Achieve an initial minimum spanning tree (MST) by performing PRIM’s algorithm on the point set (i.e. BPRIM with ε = ∞)

Create a tree from the MST by designating the first node as the source

Perform the depth first tour with a recursive function

As new edges are added to the MST, immediately remove the node’s old parent edge
Let’s start with a small set of nodes at $\varepsilon = 0.5$

Results:
- Grid size: 10 x 10
- Number of nodes: 10
- Total Wire length: 46
- Maximum Radius: 11
- Total Runtime: 0.917 ms
BRBC
Demo with small point set

How would the graphs differ if we used zero or infinity with this small point set?

$\varepsilon = 0$

Total Wire length: 53
Maximum Radius: 11
Total Runtime: 0.936 ms

$\varepsilon = \infty$

Total Wire length: 28
Maximum Radius: 17
Total Runtime: 0.926 ms
Now we perform BPRIM with a medium set of nodes at $\varepsilon = 0.5$

Results:
- Grid size: 50 x 50
- Number of nodes: 50
- Total Wire length: 1065
- Maximum Radius: 90
- Total Runtime: 11.404 ms
Lastly, we perform BPRIM with a large set of nodes at $\varepsilon = 0.5$

Results:
- Grid size: 100 x 100
- Number of nodes: 100
- Total Wire length: 2431
- Maximum Radius: 129
- Total Runtime: 78.215 ms
BPRIM vs. BRBC

Maximum Radius

Number of Nodes

- BPRIM
- BRBC
BRBC is definitely worth the extra effort on circuits with a small number of nodes. It gives a minor radius benefit at a small cost to wirelength.

BPRIM is preferable for large circuits to maintain a reasonable wirelength. Radius can be managed by reducing $\varepsilon$.

For BRBC, runtime increase dramatically as the number of nodes increases.
## Conclusions & Comparisons

<table>
<thead>
<tr>
<th></th>
<th>BPRIM</th>
<th>BRBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wirelength</td>
<td>😊</td>
<td></td>
</tr>
<tr>
<td>Radius</td>
<td></td>
<td>😊</td>
</tr>
<tr>
<td>Run Time</td>
<td>😊</td>
<td></td>
</tr>
</tbody>
</table>
Questions?

Lim, Sung Kyu, “Steiner Routing” Lecture Slides
http://users.ece.gatech.edu/limsk/course/ece6133/slides/steiner.pdf

Lim, Sung Kyu, “Practical Problems in VLSI Physical Design Automation”
http://users.ece.gatech.edu/limsk/book/