Multi net routing: Steiner Min/Max Trees

Sowmiya Sivanandham
Vidyasagar Mukala
Presentation Outline

• Overview of SMMT algorithm
• SMMT Phase – Prim’s algorithm
• SP Phase – Floyd’s algorithm
• Implementation details
• Experimental Results
• Demo
Overview of SMMT

- Routes multiple nets one-by-one
- Rip-up and re-routing
- Order in which nets are routed is important
- Given an undirected, edge-weighted graph $G(V,E)$ and a net $n$ that contains a subset of nodes $D$, SMMT of $n$ is a Steiner tree of $n$, where maximum weight among all edges in the tree is minimized
- Edge weight is equivalent to usage which reflects congestion
SMMT Cont’d

– SMMT phase (Reduces the maximum weight among all edges)
  For each un-routed net from list of ordered nets,
  • Build MST for the routing graph
  • Remove 1-degree Steiner nodes which results in SMMT
  • If wirelength of SMMT is less than \((c_j \times \text{HPBB})\), update the routing graph with the SMMT, else discard it.
  Multiple passes helps routing nets that failed in the previous passes

– SP Phase (Reduces wire length)
  For each net from list of ordered nets,
  • Rip-up from routing graph
  • Build Shortest Path tree between nodes of the net
  • If wirelength of SP tree is less than wirelength of SMMT, accept SP tree, else discard it.
MST – Prim’s Algorithm

Consider weighted graph $G = (V, E)$

Begin with $U = \{1\}$

Add one edge from $(V-U)$ to $U$ at a time

Find the shortest edge that connects $U$ and $(V-U)$ and add the vertex to $U$

Repeat until $U = V$
Implementation – Prim’s

Graph is represented using Adjacency Matrix
C[u][v] - Cost of going from node ‘u’ to node ‘v’
Lowcost[x] - Contains lowest cost through which nodes in U-V are connected to U
Closest[x] – Contains the node in {U} that is closest to node‘x’

Initialize Lowcost[i] = C[1,i]
Closest[i] = 1 // U = {1}
min = Min (Lowcost (2 to n)) //say kth node has Low cost

Add node k to U and make Lowcost[k] = ∞

Update Lowcost and Closest Arrays
for  (j = 2 to n)
    If ( C[k,j] < Lowcost[j] )
        Lowcost[j] = C[k,j]
        Closest[j] = k
end
Prim’s Algorithm - Example

<table>
<thead>
<tr>
<th>C[u][v]</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>∞</td>
<td>5</td>
<td>∞</td>
<td>3</td>
<td>∞</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>5</td>
<td>∞</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>∞</td>
<td>5</td>
<td>∞</td>
<td>∞</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>∞</td>
<td>3</td>
<td>6</td>
<td>∞</td>
<td>∞</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>∞</td>
<td>∞</td>
<td>4</td>
<td>2</td>
<td>∞</td>
<td>6</td>
</tr>
</tbody>
</table>

Graph (V,E)
### Example Cont’d

<table>
<thead>
<tr>
<th>U = ${1}$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowcost</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>Closest</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

$\min = 1; \ k = 3$

<table>
<thead>
<tr>
<th>U = ${1,3}$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowcost</td>
<td>5</td>
<td>$\infty$</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>Closest</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

$\min = 4; \ k = 6$

<table>
<thead>
<tr>
<th>U = ${1,3,6}$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowcost</td>
<td>5</td>
<td>$\infty$</td>
<td>2</td>
<td>6</td>
<td>$\infty$</td>
</tr>
<tr>
<td>Closest</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

$\min = 2; \ k = 4$
## Example Cont’d

<table>
<thead>
<tr>
<th>( U = {1,3,6,4} )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowcost</td>
<td>5</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>6</td>
<td>( \infty )</td>
</tr>
<tr>
<td>Closest</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

\( \text{min} = 5 \; ; \; k = 2 \)

<table>
<thead>
<tr>
<th>( U = {1,3,6,4,2} )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowcost</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>3</td>
<td>( \infty )</td>
</tr>
<tr>
<td>Closest</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

\( \text{min} = 3 \; ; \; k = 5 \)
Example Cont’d

<table>
<thead>
<tr>
<th>U = {1,3,6,4,2,5}</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowcost</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>Closest</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

MST of Graph (V,E)
SP – Floyd’s Algorithm $O(n^3)$

All pairs Shortest Path Solution
C[u][v] - Represents the cost of going from node ‘u’ to node ‘v’
A[u][v] – Shortest distance to go from node ‘u’ to ‘v’
P[u][v] – Stores information of intermediate node connecting node ‘u’ and node ‘v’

for (i = 1 to n)
    for(j = 1 to n)
        A[i][j] = C[i][j]
        P[i][j] = 0
    end
end

for(i = 1 to n)
    A[i][i] = 0
end

for (k = 1 to n)
    for (i = 1 to n)
        for(j = 1 to n)
                P[i][j] = k
            end
        end
    end
end
Implementation Details

Adjacency Matrix for Routing Grid

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>2</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>3</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>5</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>6</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>7</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>8</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>9</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>10</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
</tbody>
</table>
Implementation Cont’d

• Use of 3 states for weights:
  – INF → Edge cannot exist
  – 0 → Edge can exist but no net has been routed
  – n → Edge can exist and has nets routed along that edge

• Accessing each node in the routing grid
  – \( k = x \times \text{MAX} + y \)
    Where,
    • (x,y) -> co-ordinate of the vertex
    • MAX -> Maximum size of routing grid

• Size of routing grid
  – Use of MAX and actual_MAX (Static arrays)
## Experimental Analysis

<table>
<thead>
<tr>
<th>Number of nets</th>
<th>Number of terminals</th>
<th>Grid size</th>
<th>Wire length</th>
<th>Maximum Edge usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>31</td>
<td>8</td>
<td>71</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>67</td>
<td>10</td>
<td>258</td>
<td>5</td>
</tr>
<tr>
<td>30</td>
<td>87</td>
<td>13</td>
<td>313</td>
<td>7</td>
</tr>
<tr>
<td>50</td>
<td>133</td>
<td>16</td>
<td>465</td>
<td>18</td>
</tr>
</tbody>
</table>
Inference

• Routing results are dependent on the constraint $c_j$
  – Smaller $c_j \rightarrow$ tighter constraint $\rightarrow$ Routing efficiency decreases

• Congestion is dependent on grid size
  – Smaller grid with more nets increases congestion (increases maximum edge weight)
  – SMMT tries to reduce the usage (reduces congestion)
  – SP tries to reduce wire length at the cost of usage (increases congestion)
  – Multiple passes increases routing efficiency at the cost of runtime
    • During each pass, the underlying routing grid is different, although the same net is routed again
    • Hence, those nets for which routing failed in the previous passes also become routable
Routing Efficiency Vs Passes (cj = 1)

- **SMMT Pass 1:**
  - Routing in SMMT phase for net 0: **not accepted**
  - Routing in SMMT phase for net 1: **not accepted**
  - Routing in SMMT phase for net 2: **not accepted**
  - Routing in SMMT phase for net 3: **not accepted**
  - Routing in SMMT phase for net 4: accepted
  - Routing in SMMT phase for net 5: **not accepted**
  - Routing in SMMT phase for net 6: **not accepted**
  - Routing in SMMT phase for net 7: accepted
  - Routing in SMMT phase for net 8: **not accepted**
  - Routing in SMMT phase for net 9: **not accepted**

- **SMMT Pass 2:**
  - Routing in SMMT phase for net 0: **not accepted**
  - Routing in SMMT phase for net 1: **not accepted**
  - Routing in SMMT phase for net 2: **not accepted**
  - Routing in SMMT phase for net 3: **not accepted**
  - Routing in SMMT phase for net 4: accepted
  - Routing in SMMT phase for net 5: **not accepted**
  - Routing in SMMT phase for net 6: accepted
  - Routing in SMMT phase for net 7: accepted
  - Routing in SMMT phase for net 8: accepted
  - Routing in SMMT phase for net 9: **not accepted**
SMMT Pass 3:
Routing in SMMT phase for net 0: **not accepted**
Routing in SMMT phase for net 1: **not accepted**
Routing in SMMT phase for net 2: **not accepted**
Routing in SMMT phase for net 3: **not accepted**
Routing in SMMT phase for net 4: accepted
Routing in SMMT phase for net 5: **not accepted**
Routing in SMMT phase for net 6: accepted
Routing in SMMT phase for net 7: accepted
Routing in SMMT phase for net 8: accepted
Routing in SMMT phase for net 9: **not accepted**

SP Phase:
Routing in SMMT phase for net 0: **not accepted**
Routing in SMMT phase for net 1: accepted
Routing in SMMT phase for net 2: **not accepted**
Routing in SMMT phase for net 3: accepted
Routing in SMMT phase for net 4: accepted
Routing in SMMT phase for net 5: accepted
Routing in SMMT phase for net 6: accepted
Routing in SMMT phase for net 7: accepted
Routing in SMMT phase for net 8: accepted
Routing in SMMT phase for net 9: accepted
Routing Efficiency Vs Passes (cj = 3)

**SMMT Pass 1:**
Routing in SMMT phase for net 0: accepted
Routing in SMMT phase for net 1: **not accepted**
Routing in SMMT phase for net 2: accepted
Routing in SMMT phase for net 3: accepted
Routing in SMMT phase for net 4: accepted
Routing in SMMT phase for net 5: accepted
Routing in SMMT phase for net 6: accepted
Routing in SMMT phase for net 7: accepted
Routing in SMMT phase for net 8: accepted
Routing in SMMT phase for net 9: accepted

**SMMT Pass 2:**
Routing in SMMT phase for net 0: accepted
Routing in SMMT phase for net 1: **not accepted**
Routing in SMMT phase for net 2: accepted
Routing in SMMT phase for net 3: accepted
Routing in SMMT phase for net 4: accepted
Routing in SMMT phase for net 5: accepted
Routing in SMMT phase for net 6: accepted
Routing in SMMT phase for net 7: accepted
Routing in SMMT phase for net 8: accepted
Routing in SMMT phase for net 9: accepted
**Illustration Cont’d**

**SMMT Pass 3:**
Routing in SMMT phase for net 0: accepted
Routing in SMMT phase for net 1: **not accepted**
Routing in SMMT phase for net 2: accepted
Routing in SMMT phase for net 3: accepted
Routing in SMMT phase for net 4: accepted
Routing in SMMT phase for net 5: accepted
Routing in SMMT phase for net 6: accepted
Routing in SMMT phase for net 7: accepted
Routing in SMMT phase for net 8: accepted
Routing in SMMT phase for net 9: accepted

**SP Phase:**
Routing in SMMT phase for net 0: accepted
Routing in SMMT phase for net 1: accepted
Routing in SMMT phase for net 2: accepted
Routing in SMMT phase for net 3: accepted
Routing in SMMT phase for net 4: accepted
Routing in SMMT phase for net 5: accepted
Routing in SMMT phase for net 6: accepted
Routing in SMMT phase for net 7: accepted
Routing in SMMT phase for net 8: accepted
Routing in SMMT phase for net 9: accepted
### SMMT Vs SP

<table>
<thead>
<tr>
<th>NET</th>
<th>MAX E WGT</th>
<th>WIRELENGTH</th>
<th>MAX E WGT</th>
<th>WIRELENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>N/R</td>
<td>1</td>
<td>N/R</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>N/R</td>
<td>2</td>
<td>17</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>N/R</td>
<td>1</td>
<td>N/R</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>N/R</td>
<td>2</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>N/R</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>25</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>N/R</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>
Results – Phase I (SMMT)

SMMT - Phase I Net #1
(1,1) (2,2) (4,0) (4,4)

(20 ) (21 ) (22 ) (23 ) (24 )

(15 ) (16 ) (17 )-- 1--(18 )-- 1--(19 )

(10 ) (11 ) (12 ) (13 ) (14 )

(5 )-- 1--(6 ) (7 ) (8 ) (9 )

(0 )-- 1--(1 )-- 1--(2 )-- 1--(3 )-- 1--(4 )

Net 1 result after Phase I
Results – Phase II (SP)

Net 1: WL of SP(9) is less than WL of MST(12)

SHORTEST PATH – PHASE II

(20)  (21)  (22)  (23)  (24)

(15)  (16)  (17)  (18)  (19)

(10)  (11)  (12)  1--(13)  1--(14)

(5)  (6)  1--(7)  (8)  (9)

(0)  (1)  (2)  1--(3)  1--(4)

Net 1 result after Phase II
Results – Final Grid

Final Grid
Thank You!