An Efficient Non-Tree Clock Routing Algorithm for Reducing Delay Uncertainty

Morteza Saheb Zamani, Maryam Taajobian, Mehdi Saeedi
Computer Engineering Department, Amirkabir University of Technology
Tehran, Iran
{szamani, taajobian, msaedi}@aut.ac.ir

Abstract

The design of clock distribution networks in synchronous digital systems presents great challenges. In other words, controlling the clock signal delay in the presence of process parameter variations is a major problem in the design of high-speed synchronous circuits. In this paper, an efficient algorithm is presented to improve the tolerance of a clock distribution network against process variation. This algorithm generates a non–tree clock network for reducing uncertainty of the clock signal delay introduced by process variation. The non-tree clock network is generated by inserting cross links in appropriate points taking critical paths into account. To evaluate the effectiveness of the proposed algorithm several HSPICE-based Monte Carlo simulations were done. Experimental results show that the proposed approach can lead to a significant reduction in the deviation of signal propagation delay of critical paths and the skew variability without considerable increase in the total wirelength.

Track Area: Digital System Design

Topic Area: System, hardware and embedded software synthesis

Keywords: Clock distribution network, Delay uncertainty, Non-tree clock routing algorithm

1. Introduction

Clock distribution network design is one of the critical steps toward high performance, ultra deep-submicron VLSI integrated circuits. For the clock signal $c$, the following equations must be held:

$$t_c > t_f + t_i + t_s + t_{skew} \quad (1)$$

where $t_c$ is the clock period, $t_f$ is the delay of the slowest combinational logic between two synchronizing elements, $t_{skew}$ is the clock skew, $t_i$ is the set up time of the synchronizing elements and $t_s$ is their propagation delay. Since almost all data transfers in synchronized systems are coordinated by the clock signal, the main objective of clock network design is to achieve zero or bounded non-zero skew with minimum wirelength.

On the other hand, the minimum feature size of integrated circuits is shrinking and the clock frequency is increasing over the years. As a result, process parameter variations influence circuit performance significantly. Furthermore, inevitable process variation changes the quality of general signals as well as the propagation delay of the clock signal with considerable effects on the clock skew variability. The deviation of signal propagation delay from its target value, i.e. delay uncertainty $0$, is very important as its effects on the propagation delay of the clock signal can cause violations in set up and hold timing constraints of data path registers.

In this paper, a non-tree clock routing algorithm for reducing delay uncertainty induced by process variations is proposed which considers both physical location of clock sinks and critical paths. The rest of the paper is organized as follows: basic concepts are explained in Section 0. Previous work is reviewed in Section 0. The proposed algorithm is described in Section 0. Section 0 shows the experimental results and finally, Section 0 concludes the paper.

2. Basic Concepts

A non-tree RC network can be represented by a graph $G=(V,E)$ with the node set $V$ including a source, several sinks and internal nodes and the edge set $E$ representing interconnects. The graph is usually divided into a spanning tree $T=(V,E_t)$ and a set of link edges $E_l$ where $E=E_t \cup E_l$. 
The π Elmore Delay at node $i$ in an RC network is given by $t_i = \sum_j R_j C_j$, where $C_j$ is the ground capacitance at node $j$. The transfer resistance $R_{ij}$ is equal to the voltage at node $i$ when 1A current is injected into node $j$ and all other capacitors are set to be zero [17]. The final delay of node $i$ is computed by starting with delay in the tree $T$ and then incrementally adding the links and updating [11]. Consider two intermediate nodes $u$ and $w$ and a cross link inserted between them shown in Figure 1.

![Figure 1- Cross link insertion](image)

Let the link have a wire resistance of $R_l$ inserted between $u$ and $w$ and a wire capacitance $C_l$ divided into two $C_l/2$ to be added to the capacitance of $u$ and $w$ using π Elmore delay model. While it has been shown that adding the link capacitances does not change the network topology [12], [13], it changes the nominal skew of the sinks as discussed below. Suppose that before link insertion, the Elmore delay from the source to any sink $i$ is denoted as $t_i$. It can be seen that adding link capacitance changes the delay to [12]:

$$t_i = t_i + \frac{C_l}{2}(R_{uw} + R_{iu})$$

(2)

According to [11], after link insertion the delay at node $i$ is changed from $\hat{t}_i$ to $\tilde{t}_i$ as shown in Eq. (3) where $r_i$, $r_u$ and $r_w$ are equal to the Elmore delay at $i$, $u$ and $w$, respectively. In addition, the values of $C_u$ and $C_w$ are set to $1$ and $-1$, respectively and other node capacitances are also set to zero. Furthermore, $\hat{t}_i$, $\tilde{t}_u$ and $\tilde{t}_w$ are the Elmore delays at nodes $i$, $u$ and $w$, respectively after adding the link capacitance.

$$\tilde{t}_i = \hat{t}_i - \frac{t_u - t_w}{R_i + r_u - r_w} r_i$$

(3)

The skew variation between link endpoints $u$ and $w$ after link insertion is given by Eq. (4) [11] where $q_{u,w} = t_u - t_w$ is the initial skew between nodes $u$ and $w$, $q_{u,w}$ is the final skew after the link insertion and $R_{u,i}$ and $R_{w,i}$ are the transfer resistances of nodes $u$ and $w$, respectively.

$$q_{u,w} = R_i \left(\frac{R_l}{R_i + r_u - r_w} + \frac{C_l}{2}(R_{u,i} - R_{w,i})\right)$$

(4)

Since, the link capacitance $C_l$ usually changes the nominal skew, it is possible to consider only the link resistance $R_l$ and ignore $C_l$. This leads to Eq. (5) [12][13]:

$$q_{u,w} = \frac{R_i}{R_i + r_u - r_w} q_{u,w}$$

(5)

In addition, the skew between two arbitrary nodes $i$ and $j$ after inserting a link between nodes $u$ and $w$ is [12]:

$$q_{i,j} = q_{i,j} + \frac{C_l}{2}(R_{u,i} - R_{u,j} - R_{w,i} + R_{w,j}) - R_{i,j}$$

(6)

Therefore, the effect of inserting a link between nodes $u$ and $w$ on the skew between any two arbitrary nodes $i$ and $j$ can be calculated as Eq. (7).

$$q_{i,j} = q_{i,j} - \frac{r_i - r_j}{R_i + r_u - r_w} q_{u,w}$$

(7)

If a link is inserted between two nodes $u$ and $w$, the effect of $R_l$ on $q_{i,j}$ depends on the locations of nodes $i$ and $j$ in $T$ [12]. It can be said that for a given pair $i$ and $j$ in a clock network, if either $i$ or $j$ locates in a sub-tree after inserting a link between them, link insertion can reduce skew variability [12]. Consequently, in order to reduce skew variability between any node pair, it is better to distribute links over all sub-trees of the network.

3. Previous Work

In order to reduce the effects of process variations, numerous clock routing methods have been proposed [4]-[11]. The authors of [1] proposed the exact zero skew algorithm. In [2] DME algorithm was proposed to achieve minimum wire length zero skew clock. On the other hand, some of the proposed approaches as [4]-[7] and [10] allocated skew safety margins to schedule clock skew in the circuit. In [4], the skew of each clock sink pair was scheduled targeting at the middle of the skew permissible range. However, the locations of sinks were not considered in this paper. This may lead to an increase in the clock period.
Some authors as in [5] used DME method to improve the robustness of clock skew while minimizing wirelength. In other words, during the intermediate steps of the DME-based tree construction, the worst case bound of the wire delays and clock skew were analyzed and used. The algorithm in [8] generated a clock tree topology to minimize the uncertainty of the clock signal delay for the most critical data paths. However, in this paper the physical locations of clock sinks were not considered. In [12] and [13], a non-tree topology was used to reduce the skew variability via cross links as the clock signal propagating through multiple paths can reduce the variations of the circuit. However, in these methods skew variability was reduced for all sinks without considering the critical data paths which could waste resources, considerably.

4. Proposed Algorithm

To reduce delay uncertainty the authors of [8] proposed an algorithm where a clock tree topology is generated in terms of critical path tolerance to process variation. Furthermore, this algorithm increased the common parts of the clock tree from a source to critical path registers. However, this algorithm does not use the physical locations of sinks.

On the other hand, non-tree clock networks as in [12], [13] are more appropriate to reduce the skew variations. While these algorithms used both physical locations and the length of any links inserted in the clock tree, they reduced the skew variations in all paths without consideration of the circuit critical paths.

To address the problem, in this paper a new approach is presented to insert several links in the clock routing tree considering both advantages. In other words, in this paper links are inserted between two nodes where at least one critical path exists in their child. As a result, after finding all critical paths and their tolerances to the process variations, the proposed link insertion algorithm is used. In addition, as several non-effective/effective links may be deleted/inserted in the link insertion algorithm, two link selection and link insertion methods are used. Subsequently, the inserted links are added to the clock tree to improve the skew variations and clock delay uncertainty.

4.2. Link Selection Algorithm

A small difference in the clock signal delay, particularly in the clock signals of critical data path, can violate timing constraints and result in circuit malfunction. Since a clock signal propagating through multiple paths can compensate some paths variations, links could be inserted in critical paths to reduce skew variations of critical data path sinks as done in the proposed algorithm.

In this paper, the selected links between two nodes are selected such that they have at least one critical path in their child. For instance, a link between two nodes u and w is inserted in Figure 1 as there is a critical path in their child. In addition, the skew variability between two sinks becomes smaller when a link is moved from a nearest common ancestor toward leaf nodes. In other words, common portions of the two paths from the clock tree source to the sinks can be increased and the effects of parameter variations on the common part introduces identical delays to those clock signals driving sequentially adjacent registers [8]. As a result, for deeper links the common parts of the clock signal reduce the skew variations more efficiently. Bipartite graph was used in this paper to distribute the inserted links throughout the clock network as well as control the number of inserted links and reduce the skew variations of all sinks. The proposed link selection algorithm and its subroutine are shown in Figure 2 and Figure 3, respectively.

The subroutine shown in Figure 2 constructs a bipartite graph of the initial tree. In each iteration, the subtree \( T_v \) is decomposed into its left \( (T_l) \) and right \( (T_r) \) children. Then some node pair sets between \( T_l \) and \( T_r \) are characterized using the subroutine shown in Fig. 3. Since binary zero skew clock tree is used in this paper, the links are distributed throughout the clock network by applying this method.

If the depths of links are large, the sinks and deep nodes may be far from each other. On the other hand, common parts of the two paths in short links are not increased greatly. Our simulations show that if the depths of links are greater than half of the tree depth, they are suitable choices.

To select pair sets between nodes, the subtrees \( T_l \) and \( T_r \) are decomposed into k subtrees. Then, a node pair is selected if it has a critical path in its children and its depth is sufficient.

**Procedure: Select_Node_Pairs(T_v)**

**Input:** Subtree \( T_v \) rooted at node \( v \)  
**Output:** Node pair set \( P \)

1. \( l \leftarrow \) left child node of \( v \)  
2. \( r \leftarrow \) right child node of \( v \)  
3. Pair_Between_Trees \( (T_l, T_r) \)  
4. If Depth \( (v) \) = MaxDepth return \( P \)  
5. \( P \leftarrow \) Select_Node_Pairs \( (T_l) \)  
6. \( P \leftarrow \) Select_Node_Pairs \( (T_r) \)  
7. Return \( P \)

**Figure 2- The link selection algorithm**
Subroutine: Pair_Between_Trees(T₁,T₂)

Input: Two subtrees T₁ and T₂
Output: Node pair set P

1. Decompose T₁ into subtrees S₁={T₁₁, T₁₂, ..., T₁ₖ}
2. Decompose T₂ into subtrees S₂={T₂₁, T₂₂, ..., T₂₇}
3. For every pair (T₁ᵢ, T₂ᵢ) between S₁ and S₂
4. If there is a Critical Path with the following conditions
5. (a) its first node is in T₁ᵢ and its last one is in T₂ᵢ and
6. (b) Depth(T₁ᵢ)≥Depth(T₂ᵢ) and Depth(T₂ᵢ)≥Depth(T₁ᵢ)
7. Then
8. P+ Link Between root of T₁ᵢ and root of T₂ᵢ
9. Return P

Figure 3- The link selection subroutine

4.2. Link Insertion Algorithm

To improve both the skew variations and the delay uncertainty in the critical paths simultaneously, in this paper a cost function is defined as Eq. (8) to examine the following parameters in the link insertion stage:

- The length of selected links
- The skew variations
- The critical path delay

Total Delay = Data Path Delay + Clock Skew + Skew Variation  (8)

Since the initial clock tree is considered to have zero skew, the clock skew in Eq. 8 is the skew induced by link insertion and is removed in the link insertion algorithm. Therefore, Eq. (8) is reduced to Eq. (9):

Total Delay = Data Path Delay + Skew Variation  (9)

Based on the previous equations, the following cost function is defined to insert links

\[ \text{Cost Function} = \alpha f + \alpha_r \text{Length} \]

where

\[ f = \begin{cases} 0 & D < \text{timing wall} \\ (D - \text{timing wall})^2 & D > \text{timing wall} \end{cases} \]

\[ D = \text{critical path delay} + \text{skew variation} \]

where Length shows the increase in wirelength in the clock network after the link insertion phase and f is used to show the effects of the link insertion on the skew variation and critical path delay. In these equations, timing wall is calculated from maximum required frequency.

The proposed algorithm for inserting links in the clock tree network is illustrated in Figure 4. As shown in this figure, each selected link is added to the clock tree and the effect of this link is evaluated. If the added link improves the cost function, it is inserted to the selected link set.

5. Experimental Results

The proposed algorithm was implemented in C++ and all the experiments were done on an Intel Pentium IV 3GHz computer with 1GB memory. To evaluate the algorithm, we used the ISCAS89 benchmarks [15]. The characteristics of each benchmark are shown in Table 1. For our experiments, we used ±15% variation in the wire width and the wire thickness using a normal distribution. To evaluate the results, HSPICE-based Monte Carlo simulation with 1000 iterations was used.

<table>
<thead>
<tr>
<th>#</th>
<th>Benchmark</th>
<th>Total # of cells</th>
<th># of memory cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>s208_1</td>
<td>73</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>s349</td>
<td>123</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>s344</td>
<td>132</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>s444</td>
<td>162</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>s420_1</td>
<td>187</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>s838_1</td>
<td>367</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>s1196</td>
<td>483</td>
<td>18</td>
</tr>
<tr>
<td>8</td>
<td>s1423</td>
<td>607</td>
<td>74</td>
</tr>
<tr>
<td>9</td>
<td>s15850</td>
<td>685</td>
<td>165</td>
</tr>
<tr>
<td>10</td>
<td>s9234_1</td>
<td>974</td>
<td>145</td>
</tr>
<tr>
<td>11</td>
<td>s13207</td>
<td>1219</td>
<td>333</td>
</tr>
</tbody>
</table>

In order to obtain an initial clock tree, the BST code [16] was used. Furthermore, we compared the proposed critical path-based link insertion algorithm
(Link-CP) with two other recent algorithms namely common path [8] and Link-M [12]. The former generates a clock tree topology to minimize the uncertainty of the clock signal delay in critical paths and the latter is a min-matching based link insertion algorithm that reduces skew variability in all sinks by considering the physical locations of the sinks. In addition, the maximum skew variation in both Elmore delay (MSV-E) and HSPICE-based models (MSV-S) were evaluated. We also compared the standard deviation values of skew variation for both Elmore delay (SD-E) and HSPICE-based models (SD-S) in the clock network. Note that the skew variation is difference between the maximum sink delay and minimum sink delay. The delay uncertainty of critical paths and wire length of the clock networks were also compared.

Table 2 shows the MSV and SD of initial clock tree and those proposed by the non-tree (N-Tree) clock network. The results in this table indicate that for both delay models, our algorithm reduces (Red.) the MSV as well as the SD in all circuits. In addition, the values of MSV and SD in HSPICE-based model show that the resulted values are very similar to Elmore delay model.

Table 3 shows the delay uncertainty in critical paths and the amount of increase in wirelength for both the proposed non-tree and the conventional tree clock networks. It can be verified that the proposed non-tree clock topology could improve the delay uncertainty in critical paths efficiently.

Table 4 shows the comparison of the proposed Link-CP algorithm with two other link-M [12] and common path algorithms [8]. It can be seen that the common path algorithm has more MSV, SD and wirelength than the two other algorithms because it does not consider the physical locations of sinks. However, this algorithm improves the delay uncertainty compared to the Link-M algorithm. On the other hand, the Link-M algorithm has less MSV, SD and wirelength than the common path algorithm with more delay uncertainty as [12] does not construct critical paths and reduces the skew variation for all sinks.

The results of this table show that for all circuits, the delay uncertainty is reduced and there is a certain correlation between the amounts of increase in wirelength and decrease in the delay uncertainty. This table also shows that for larger circuits, delay uncertainty reductions and the wirelength increases were decreased.

<table>
<thead>
<tr>
<th>#</th>
<th>Routing Methods</th>
<th>MSV-E</th>
<th>MSV-S</th>
<th>SD-E</th>
<th>SD-S</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tree</td>
<td>59</td>
<td>44.4</td>
<td>15.5</td>
<td>17.5</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>39.7</td>
<td>24.0</td>
<td>8.3</td>
<td>10.0</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>36%</td>
<td>45%</td>
<td>48%</td>
<td>41%</td>
</tr>
<tr>
<td>2</td>
<td>Tree</td>
<td>91.7</td>
<td>115.6</td>
<td>24.6</td>
<td>77.8</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>64.1</td>
<td>90.3</td>
<td>16.7</td>
<td>34.1</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>30%</td>
<td>21%</td>
<td>33%</td>
<td>55%</td>
</tr>
<tr>
<td>3</td>
<td>Tree</td>
<td>144.9</td>
<td>154.8</td>
<td>25.1</td>
<td>39.6</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>134.5</td>
<td>133.0</td>
<td>17.7</td>
<td>18.1</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>6%</td>
<td>13%</td>
<td>32%</td>
<td>50%</td>
</tr>
<tr>
<td>4</td>
<td>Tree</td>
<td>149.2</td>
<td>145.8</td>
<td>39.6</td>
<td>38.7</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>75.9</td>
<td>100.7</td>
<td>29.4</td>
<td>27.2</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>48%</td>
<td>31%</td>
<td>25%</td>
<td>28%</td>
</tr>
<tr>
<td>5</td>
<td>Tree</td>
<td>122.9</td>
<td>179.45</td>
<td>25.2</td>
<td>48.5</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>79.6</td>
<td>140.8</td>
<td>20.6</td>
<td>25.2</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>35%</td>
<td>21%</td>
<td>20%</td>
<td>45%</td>
</tr>
<tr>
<td>6</td>
<td>Tree</td>
<td>682.3</td>
<td>575.9</td>
<td>60.7</td>
<td>192.1</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>666.7</td>
<td>540.6</td>
<td>34.2</td>
<td>140.9</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>2%</td>
<td>6%</td>
<td>43%</td>
<td>27%</td>
</tr>
<tr>
<td>7</td>
<td>Tree</td>
<td>253.9</td>
<td>256.3</td>
<td>74.9</td>
<td>123.7</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>206.5</td>
<td>204.7</td>
<td>54.1</td>
<td>105.7</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>18%</td>
<td>20%</td>
<td>27%</td>
<td>14%</td>
</tr>
<tr>
<td>8</td>
<td>Tree</td>
<td>927.3</td>
<td>958.1</td>
<td>86.1</td>
<td>180.7</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>900.4</td>
<td>865.2</td>
<td>39.0</td>
<td>180.9</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>3%</td>
<td>9%</td>
<td>54%</td>
<td>30%</td>
</tr>
<tr>
<td>9</td>
<td>Tree</td>
<td>966.5</td>
<td>2972.1</td>
<td>145.8</td>
<td>162.8</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>811.2</td>
<td>2586.6</td>
<td>122.9</td>
<td>203.0</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>16%</td>
<td>12%</td>
<td>15%</td>
<td>-25%</td>
</tr>
<tr>
<td>10</td>
<td>Tree</td>
<td>5043.4</td>
<td>3549.8</td>
<td>1025.7</td>
<td>2861.1</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>3237.2</td>
<td>1456.7</td>
<td>616.9</td>
<td>1343.9</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>35%</td>
<td>58%</td>
<td>39%</td>
<td>53%</td>
</tr>
<tr>
<td>11</td>
<td>Tree</td>
<td>2070.5</td>
<td>6645.3</td>
<td>529.6</td>
<td>1544.2</td>
</tr>
<tr>
<td></td>
<td>N-Tree</td>
<td>1760.5</td>
<td>5138.1</td>
<td>471.8</td>
<td>1979.0</td>
</tr>
<tr>
<td></td>
<td>Red.</td>
<td>14%</td>
<td>22%</td>
<td>10%</td>
<td>-28%</td>
</tr>
</tbody>
</table>

Average: 22% 22% 29% 26%
6. Conclusion

In this paper, a new non-tree clock routing algorithm was proposed to reduce the skew variability and the delay uncertainty induced by process variation. This algorithm is based on the link insertion method and considers critical paths. The effectiveness of the proposed algorithm was validated using HSPICE-based Monte Carlo simulation. Experimental results show that the proposed algorithm is able to reduce the delay uncertainty in critical paths and the skew variability with little increase in the wirelength.

References

Table 4- The comparisons of the proposed algorithm with two recent algorithms [8] and [12]

<table>
<thead>
<tr>
<th>#</th>
<th>Routing Methods</th>
<th>MSV</th>
<th>SD</th>
<th>Delay Uncertainty</th>
<th>Wirelength</th>
<th>Run Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Common Path</td>
<td>50.6</td>
<td>11.8</td>
<td>45.8</td>
<td>235.5</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>49.4</td>
<td>9.5</td>
<td>48.3</td>
<td>201.8</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>39.7</td>
<td>6.3</td>
<td>31.7</td>
<td>154.6</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>Common Path</td>
<td>137.8</td>
<td>38.1</td>
<td>61.3</td>
<td>924.9</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>91.7</td>
<td>24.6</td>
<td>70.0</td>
<td>558.0</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>64.1</td>
<td>16.7</td>
<td>50.4</td>
<td>437.2</td>
<td>0.2</td>
</tr>
<tr>
<td>3</td>
<td>Common Path</td>
<td>441.5</td>
<td>171.0</td>
<td>118.0</td>
<td>749.7</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>145.0</td>
<td>24.7</td>
<td>120.5</td>
<td>434.1</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>134.5</td>
<td>17.7</td>
<td>112.2</td>
<td>358.1</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>Common Path</td>
<td>291.4</td>
<td>65.1</td>
<td>286.4</td>
<td>823.5</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>85.5</td>
<td>24.3</td>
<td>80.0</td>
<td>616.7</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>75.9</td>
<td>9.4</td>
<td>72.2</td>
<td>537.7</td>
<td>0.5</td>
</tr>
<tr>
<td>5</td>
<td>Common Path</td>
<td>183.7</td>
<td>77.0</td>
<td>83.7</td>
<td>914.2</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>126.3</td>
<td>21.0</td>
<td>93.5</td>
<td>502.0</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>79.6</td>
<td>20.6</td>
<td>63.1</td>
<td>453.6</td>
<td>0.5</td>
</tr>
<tr>
<td>6</td>
<td>Common Path</td>
<td>825.4</td>
<td>180.5</td>
<td>635.4</td>
<td>1925.4</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>795.7</td>
<td>52.6</td>
<td>717.7</td>
<td>1516.0</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>666.7</td>
<td>34.2</td>
<td>580.7</td>
<td>1449.6</td>
<td>0.5</td>
</tr>
<tr>
<td>7</td>
<td>Common Path</td>
<td>414.3</td>
<td>125.8</td>
<td>1450.0</td>
<td>1050.0</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>285.4</td>
<td>65.2</td>
<td>161.8</td>
<td>610.1</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>206.5</td>
<td>14.1</td>
<td>131.6</td>
<td>570.0</td>
<td>0.5</td>
</tr>
<tr>
<td>8</td>
<td>Common Path</td>
<td>1501.4</td>
<td>197.4</td>
<td>951.6</td>
<td>5051.0</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>1005.0</td>
<td>1150.2</td>
<td>1003.1</td>
<td>3521.9</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>900.4</td>
<td>39.0</td>
<td>805.3</td>
<td>3083.9</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>Common Path</td>
<td>1271.8</td>
<td>226.0</td>
<td>751.0</td>
<td>5930.0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>947.1</td>
<td>115.5</td>
<td>898.0</td>
<td>4895.0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>811.2</td>
<td>42.9</td>
<td>609.1</td>
<td>4389.7</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>Common Path</td>
<td>4995.4</td>
<td>1010.8</td>
<td>1875.2</td>
<td>6014.3</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>3568.1</td>
<td>795.7</td>
<td>2022.1</td>
<td>4825.4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>3237.2</td>
<td>616.9</td>
<td>237.4</td>
<td>4574.4</td>
<td>30</td>
</tr>
<tr>
<td>11</td>
<td>Common Path</td>
<td>3560.8</td>
<td>585.6</td>
<td>1621.0</td>
<td>13531.5</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Link-M</td>
<td>2000.3</td>
<td>306.7</td>
<td>1845.8</td>
<td>9823.7</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Link-CP</td>
<td>1760.5</td>
<td>271.8</td>
<td>1255.0</td>
<td>9691.3</td>
<td>120</td>
</tr>
</tbody>
</table>


