A TIMING-DRIVEN GLOBAL ROUTER BASED ON CRITICAL NETWORK TECHNIQUE

Tong Jing, Xianlong Hong, Haiyun Bao,
Yici Cai, Jingyu Xu

Dept. of Computer Science and Technology, Tsinghua Univ.
Beijing, China
Contents

1. Introduction
2. The Critical-Network based Timing Analysis Technique
3. Our Global Routing Algorithm
4. Experimental Results
5. Conclusion and Future Work
1. Introduction

- With progress in Very-deep-submicron (VDSM) technology and giga-scale integrated circuits (GSI), Interconnect delay has become a dominant factor.
- Only congestion-driven global router dose not meet the request.
- Need to develop timing-driven global routing algorithms
Previous Work for Timing Driven Routing

- Timing models [3-6]
- Single net routing
  - Timing-driven Steiner tree algorithms [7, 11]
  - Interconnect planning algorithms considering buffer insertion or/and wire sizing [8-10, 12-14]
- Multiple nets routing
  - Timing-driven global routing algorithms [15-22]
**Net-based Timing Analysis Strategy**

- Distribute the path delay into related nets
- Is easy to be implemented and controlled
- Has the blindness in delay assignment
- Is hard to optimize other objectives
Critical-Path-based Timing Analysis Strategy

- Avoids the unreasonable delay assignment
- Relaxes timing constraints in whole path
- Obtains a better routing result
- Depends on greedy trying without any active control
- Takes much time to check all the critical paths
The main contribution of this work

Proposed a **Critical Network Concept** and timing optimizing strategy based on critical network concept.

It is different from the existing strategies. It is possible to reduce the delay in an overall view.

It is easy to combine the optimization of other objectives.
2. The Critical-Network-Based Timing Analysis Technique

- **Critical Pin:** If $t_E(i) > t_L(i)$, then, pin $i$ is called a critical pin.

  - $t_E(i)$ is the earliest arriving time of pin $i$
  - $t_L(i)$ is the latest request time of pin $i$

- **Critical Edge:** If $(i,k) \in E$, and $t_E(i) + delay(i,k) > t_L(k)$, then, edge $(i,k)$ is called a critical edge.
Four theorems

- The two pins of a critical edge are both critical pins.
- Any critical pin except the sink pin $t$ has at least one critical edge starting from it.
- Any critical pin except the source pin $s$ has at least one critical edge ending to it.
- The overtime (i.e., the earliest time is late than the latest time) of the source pin $s$ is equal to that of the sink pin $t$. 
Critical network concept

**Critical Network**

From the four theorems, we have: If $T_E > T_L$, there is a network $N = (V, E, d, s, t)$ consisting of and only consisting of critical pins and critical edges. $N$ is called a critical network.

where $V$ is the set of critical pins

$E$ is the set of critical edges

$d$ is the weight of edge $(i,k)$

$T_E$ is the earliest arriving time of $t$

$T_L$ is the longest allowable delay of the circuit
Partial critical network in MCNC C2
3. Our Global Routing Algorithm

- Problem Formulation

To construct routing trees for all nets on the global routing graph (GRG) to meet congestion and timing constraints.
Timing Model: Sakurai delay based timing model

\[ T_{DZ} = \beta R_s (c_e + C_z) + \alpha r_e c_e + \beta r_e C_z \]

- \( R_s \): on-resistance of the transistor,
- \( r_e \): resistance of distributed RC line
- \( c_e \): capacitance of distributed RC line
- \( C_z \): loading capacitance.
Congestion Minimization based Net Rerouting

Search Space Traversing Technique (SSTT):

```plaintext
FOR i := 1 TO K DO
    Call Router1;
    Call Router2;
    IF No overflow THEN EXIT
END FOR

IF Overflow THEN Call Router3;

Router1: Sequential router;
Router2: Stochastic router
Router3: Local enumeration router;
```
Timing optimizing

- Define edge weight $d'$ in critical network, which is used to evaluate the cost of improving the delay of each net, create critical network $N(V,E,d',s,t)$;
- Adopt the technique of maximum flow and min-cut for the critical network to find a set of edges, on which we reroute nets to reduce the critical path delay with less effects in routing congestion;
- Construct the routing tree with least timing cost and replace the current routing tree by the one with least timing cost to reduce the delay of a net.
Definition of Edge Weight - \( d' \)

\( T \) - current routing tree,
\( T' \) - routing tree with the least timing cost,
\( w_{T} \) - congestion value of \( T \),
\( w_{T'} \) - congestion value of \( T' \)

\* Evaluating cost of the delay improvement

\( IF (T = T') \cup (delay(T') \geq delay(T)), THEN d' = \infty; \)
\( IF (delay(T') < delay(T)) \cap (w_{T'} \leq w_{T}), THEN d' = 0; \)
\( IF (delay(T') < delay(T)) \cap (w_{T'} > w_{T}), \)
\( \quad THEN d' = (w_{T'} - w_{T})^2 / (delay(T) - delay(T')). \)

\* Using \( d' \), we create a new network \( N' = (V, E, d', s, t). \)
The Timing Driven GR Algorithm

ALGORITHM GR
S1: construct initial minimum wire length Steiner Tree for each net respectively;
S2: make statistics of total routing resources and determine congested edges;
S3: analyze critical paths and create critical network $N$;
WHILE $((T_B > T_L) \text{ OR } (\text{maxFlow} \neq \infty))$ DO
{
    S4: CALL SSTT-Router to reroute congested nets;
    S5: optimize the circuit delay;
        S51: compute $d'$ in $N$;
        S52: replace $d$ by $d'$ and create $N'$;
    S53: Compute the maximum flow of $N'$;
        IF $N'$ maxFlow = $\infty$, THEN RETURN;
        Obtain the nets corresponding to the min-cut of $N'$;
        Improve the delay of the nets corresponding to the min-cut of $N'$;
    S6: update delay information of each pin in the circuit;
    S7: update total routing resources and recalculate the GRG congestion;
}

### 4. Experimental Results

**Benchmark data**

<table>
<thead>
<tr>
<th>Circuits</th>
<th>Nets</th>
<th>Grids</th>
<th>Allowable Delay (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>745</td>
<td>9*11</td>
<td>0.496281</td>
</tr>
<tr>
<td>C5</td>
<td>1763</td>
<td>16*18</td>
<td>1.045857</td>
</tr>
<tr>
<td>C7</td>
<td>2356</td>
<td>16*18</td>
<td>1.083933</td>
</tr>
<tr>
<td>S13207</td>
<td>4953</td>
<td>24*26</td>
<td>1.022260</td>
</tr>
<tr>
<td>Avq</td>
<td>21851</td>
<td>65*67</td>
<td>4.023312</td>
</tr>
</tbody>
</table>
Comparison on delay (12 experiments)

### Table 2 Comparison on Delay (average)

<table>
<thead>
<tr>
<th>Circuits</th>
<th>T Mode Delay (ns)</th>
<th>W Mode Delay (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>0.0000 (0.00%)</td>
<td>0.0155 (+3.12%)</td>
</tr>
<tr>
<td>C5</td>
<td>-0.0029 (-0.28%)</td>
<td>0.0803 (+7.68%)</td>
</tr>
<tr>
<td>C7</td>
<td>-0.0039 (-0.36%)</td>
<td>0.0300 (+2.77%)</td>
</tr>
<tr>
<td>S13207</td>
<td>-0.0010 (-0.10%)</td>
<td>0.0711 (+6.96%)</td>
</tr>
<tr>
<td>Avq</td>
<td>-0.0116 (-0.29%)</td>
<td>0.0345 (+0.86%)</td>
</tr>
</tbody>
</table>

### Table 3 Comparison on the Greatest Delay

<table>
<thead>
<tr>
<th>Circuits</th>
<th>T Mode Delay (ns)</th>
<th>W Mode Delay (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>0.0000 (0.00%)</td>
<td>0.0440 (+8.87%)</td>
</tr>
<tr>
<td>C5</td>
<td>-0.0002 (-0.02%)</td>
<td>0.1239 (+11.85%)</td>
</tr>
<tr>
<td>C7</td>
<td>-0.0005 (-0.05%)</td>
<td>0.0895 (+8.26%)</td>
</tr>
<tr>
<td>S13207</td>
<td>0.0000 (0.00%)</td>
<td>0.1857 (+18.17%)</td>
</tr>
<tr>
<td>Avq</td>
<td>0.0000 (0.00%)</td>
<td>0.3004 (+7.47%)</td>
</tr>
</tbody>
</table>
Comparison on wire length

<table>
<thead>
<tr>
<th>Circuits</th>
<th>W Mode (µm)</th>
<th>T Mode (µm)</th>
<th>Percentage Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>4.7115*10^4</td>
<td>4.7106*10^4</td>
<td>(-0.02%)</td>
</tr>
<tr>
<td>C5</td>
<td>1.2789*10^5</td>
<td>1.2755*10^5</td>
<td>(-0.27%)</td>
</tr>
<tr>
<td>C7</td>
<td>1.5494*10^5</td>
<td>1.5567*10^5</td>
<td>(+0.47%)</td>
</tr>
<tr>
<td>S13207</td>
<td>1.0166*10^6</td>
<td>1.0157*10^6</td>
<td>(-0.09%)</td>
</tr>
<tr>
<td>Avq</td>
<td>1.2760*10^6</td>
<td>1.2757*10^6</td>
<td>(-0.02%)</td>
</tr>
</tbody>
</table>
Comparison on overflow edges

<table>
<thead>
<tr>
<th>Circuits</th>
<th>W Mode</th>
<th>T Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>0.17</td>
<td>0.25</td>
</tr>
<tr>
<td>C5</td>
<td>0.13</td>
<td>0.91</td>
</tr>
<tr>
<td>C7</td>
<td>0.00</td>
<td>0.16</td>
</tr>
<tr>
<td>S13207</td>
<td>0.38</td>
<td>0.75</td>
</tr>
<tr>
<td>Avq</td>
<td>1.38</td>
<td>1.00</td>
</tr>
</tbody>
</table>
5. Conclusion and Future Work

- There are encouraging experimental results.
- This timing-driven global routing algorithm is able to control the circuit delay efficiently.
- Causes little negative effects on other optimizing objectives, such as wire length and overflow edges.
Future work

- Take coupling effects and crosstalk into account
- Consider to combine buffer insertion and shielding
Thank you!
\[ t_E(k) = \begin{cases} 
0 & k = s; \\
\max(t_E(i) + \text{delay}(i, k) \mid (i, k) \in E) & \text{otherwise}
\end{cases} \]

\[ t_L(i) = \begin{cases} 
T_L & i = t; \\
\min(t_L(k) - \text{delay}(i, k) \mid (i, k) \in E) & \text{otherwise}
\end{cases} \]