UCLA TRIO Package

Jason Cong, Lei He
Cheng-Kok Koh, and David Z. Pan

UCLA Computer Science Dept
Los Angeles, CA 90095
Optimal Interconnect Synthesis

Constraints:
• Delay
• Skew
• Signal Integrity
...

Optimized Interconnect designs:

- Sizing
- Topology
- Spacing

- Automatic solutions guided by accurate interconnect models
Technology advances lead to the need for interconnect-driven design

Interconnect optimization techniques for performance and signal integrity
- Topology optimization
- Buffer(repeater) insertion
- Device sizing, wire sizing and spacing

TRIO: Tree, Repeater, and Interconnect Optimization

Goal: to develop a unified framework to apply various interconnect layout optimization techniques independently or simultaneously
Components of TRIO

- **Optimization engine**
  - Tree construction
  - Buffer (repeater) insertion
  - Device sizing, wire sizing and spacing

- **Delay computation**
  - Elmore delay model
  - Higher-order delay model

- **Device delay and interconnect capacitance model**
  - Simple formula-based model
  - Table look-up based model
Optimization Engines of TRIO

- **Tree construction**
  - A-tree, buffered A-tree, and RATS-tree

- **Buffer insertion**

- **Wire sizing and spacing**
  - Single-source wire sizing
  - Multi-source wire sizing
  - Global wire sizing and spacing with coupling cap

- **Simultaneous device and interconnect optimization:**
  - Simultaneous buffer insertion and wire sizing
  - Simultaneous device and wire sizing
    - simple models for device delay and interconnect cap
  - Simultaneous device sizing, and wire sizing and spacing
    - table-based models for device delay and coupling cap
Classification of TRIO Algorithms

- **Bottom-up approach**
  - A-tree [Cong-Leung-Zhou, DAC’93]
  - Buffered and wiresized A-tree [Okamoto-Cong, ICCAD’96]
  - RATS-tree [Cong-Koh, ICCAD’97]
  - Simultaneous buffer insertion and wire sizing [Lillis-Cheng-Lin, ICCAD’95]
  - Global interconnect sizing and spacing with coupling cap [Cong-He-Koh-Pan, ICCAD’97]

- **Local-refinement (LR) based approach**
  - Single-source wire sizing [Cong-Leung, ICCAD’93]
  - Multi-source and variable-segmentation wire sizing [Cong-He, ICCAD’95]
  - Simultaneous driver/buffer and wire sizing [Cong-Koh, ICCAD’94, Cong-Koh-Leung, ISLPED’96]
  - Simultaneous device sizing, and wire sizing and spacing using table-based models for device delay and coupling cap [Cong-He, ICCAD’96, TCAD’99]
A-tree Algorithm [Cong-Leung-Zhou, DAC’93]

- A-tree: Rectilinear Steiner arborescence (shortest path tree)
- Resistance ratio: Driver resistance vs. unit wire resistance
- As resistance ratio decreases, min-cost A-tree has better performance than Steiner minimal tree
- A-tree algorithm
  - Start with a forest of n single-node A-trees, repeatedly
    - Grow an existing A-tree, or
    - Combine two A-trees into a new one
Buffer Insertion Algorithm
[van Ginneken, ISCAS’90]

- Given topology, buffer types, and candidate buffer locations, insert buffers to minimize maximum sink delay
Optimal Buffer Insertion by Dynamic Programming

- Bottom-up computation of irredundant set of options (c,q)'s at each buffer candidate location

- Option (c,q),
  - c: Cap. of DC-connected subtree
  - q: Req. arrival time corresponding to c

- Pruning Rule: For (c,q) and (c’, q’), (c’, q’) is redundant if c’ ≥ c and q’ < q

- Total number of options in the source is polynomial-bounded

- Top-down selection of optimal buffer types and buffer locations
Further Works on Bottom-up Approach

- Simultaneous buffer insertion and wire sizing [Lillis-Chen-Lin, ICCAD’95]
- **Wiresized Buffered A-tree (WBA-tree)** [Okamoto-Cong, ICCAD’96]
  - Combination of A-tree, simultaneous buffer insertion and wire sizing
- Global interconnect sizing and spacing considering coupling cap [Cong-He-Koh-Pan, ICCAD’97]
- **RATS-tree** [Cong-Koh, ICCAD’97]
  - Extension to higher-order delay model via bottom-up moment computation
Classification of TRIO Algorithms

- **Bottom-up approach**
  - A-tree [Cong-Leung-Zhou, DAC’93]
  - Buffered and wiresized A-tree [Okamoto-Cong, ICCAD’96]
  - RATS-tree [Cong-Koh, ICCAD’97]
  - Simultaneous buffer insertion and wire sizing [Lillis-Cheng-Lin, ICCAD’95]
  - Global interconnect sizing and spacing with coupling cap [Cong-He-Koh-Pan, ICCAD’97]

- **Local-refinement (LR) based approach**
  - Single-source wire sizing [Cong-Leung, ICCAD’93]
  - Multi-source and variable-segmentation wire sizing [Cong-He, ICCAD’95]
  - Simultaneous driver/buffer and wire sizing [Cong-Koh, ICCAD’94, Cong-Koh-Leung, ISLPED’96]
  - Simultaneous device sizing, and wire sizing and spacing using table-based models for device delay and coupling cap [Cong-He, ICCAD’96, TCAD’99]
Discrete Wiresizing Optimization
[Cong-Leung, ICCAD’93]

- **Given:** A set of possible wire widths \( \{ W_1, W_2, \ldots, W_r \} \)
- **Find:** An optimal wire width assignment to minimize weighted sum of sink delays
Dominance Relation and Local Refinement
[Cong-Leung, ICCAD 93]

- **Dominance Relation**
  
  For all \( E_j, \quad W(E_j) \geq W'(E_j) \)

  \( \Downarrow \)

  \( W \) dominates \( W' \)

- **Local Refinement of \( E \)**

  Given wire width assignment \( W \) compute optimal wire width of \( E \) assuming other wire width fixed in \( W \)
Dominance Property for Optimal Wiresizing

- Theorem (Dominance Property):
  - Assignment $W$ dominates optimal assignment $W^*$
    - $W' = \text{local refinement of } W$
    - Then, $W'$ dominates $W^*$
  - If $W$ is dominated by $W^*$
    - $W' = \text{local refinement of } W$
    - Then, $W'$ is dominated by $W^*$

- Application of Dominance Property
  
  $W_0 = \text{Min Width Assignment (dominated by opt. sol.)}$
  $W_1 = \text{Local-Refinement}(W_0)$
  $W_2 = \text{Local-Refinement}(W_1)$

  - $W_i$ dominated by opt. sol. $\Rightarrow$ lower bound computation
Further Works on LR-based Approach

- Multi-source wire sizing optimization with variable segmentation [Cong-He, ICCAD’95]
  - Bundled local refinement (BLR) that is 100x faster than local refinement (LR)

- Simultaneous driver/buffer and wire sizing [Cong-Koh, ICCAD’94, Cong-Koh-Leung, ISLPED’96]

- Simultaneous device and wire sizing [Cong-He, PDW’96, ICCAD’96]

- General case: extended local refinement (ELR) for three classes of CH-programs [Cong-He, ISPD’98, TCAD’99]
  - e.g., simultaneous device sizing, wire sizing and spacing under table models rather than simple models in most works

- LR to minimize maximum delay via Lagrangian Relaxation [Chen-Chang-Wong, DAC’96]
Table-based Model for Device

<table>
<thead>
<tr>
<th>effective-resistance $R_0$ for unit-width n-transistor</th>
</tr>
</thead>
<tbody>
<tr>
<td>size = 100x</td>
</tr>
<tr>
<td>$c_1 \backslash t_t$</td>
</tr>
<tr>
<td>0.225pf</td>
</tr>
<tr>
<td>0.425pf</td>
</tr>
<tr>
<td>0.825pf</td>
</tr>
<tr>
<td>size = 400x</td>
</tr>
<tr>
<td>$c_1 \backslash t_t$</td>
</tr>
<tr>
<td>0.501pf</td>
</tr>
<tr>
<td>0.901pf</td>
</tr>
<tr>
<td>1.701pf</td>
</tr>
</tbody>
</table>

- $R_0$ depends on size, input transition time and output loading
  - Neither a constant nor a function of a single variable
  - Device sizing problem no longer has a unique local optimum

- Lower and upper bounds of exact solution can be computed by ELR operation [Cong-He, ISPD’98, TCAD’99]
Experiment Results:

- **SPICE-delay comparison**
  - sgws: LR-based simultaneous gate and wire sizing
  - stis: LR-based simultaneous transistor and wire sizing

<table>
<thead>
<tr>
<th></th>
<th>simple-model</th>
<th>table-model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DCLK</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sgws</td>
<td>1.16 (0.0%)</td>
<td>1.08 (-6.8%)</td>
</tr>
<tr>
<td>stis</td>
<td>1.13 (0.0%)</td>
<td>0.96 (-15.1%)</td>
</tr>
<tr>
<td><strong>2cm line</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sgws</td>
<td>0.82 (0.0%)</td>
<td>0.81 (-0.4%)</td>
</tr>
<tr>
<td>stis</td>
<td>0.75 (0.0%)</td>
<td>0.69 (-7.6%)</td>
</tr>
</tbody>
</table>

- **Runtime**
  - Total LR-based optimization: ~10 seconds
  - Total HSPICE simulation: ~3000 seconds

- **Manual optimization of DCLK**
  - delay is 1.2x larger, and power is 1.3x higher
Global Interconnect Sizing and Spacing (GISS)

- **SISS**: Single-net interconnect sizing and spacing
- **GISS**: Global interconnect sizing and spacing
  - **GISS/DP**: Bottom-up based approach [Cong-He-Koh-Pan, ICCAD’97]
  - **GISS/ELR**: ELR based approach [Cong-He, ISPD’98, TCAD’99]
- All use table-based capacitance model with coupling capacitance [Cong-He-Kahng-et al, DAC’97]
## Experiment Results

<table>
<thead>
<tr>
<th>Center spacing</th>
<th>Average Delays (ns)</th>
<th>Runtimes (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SISS</td>
<td>GISS/DP</td>
</tr>
<tr>
<td>1.10um</td>
<td>1.31</td>
<td>0.79 (-39%)</td>
</tr>
<tr>
<td>1.65um</td>
<td>0.72</td>
<td>0.53 (-26%)</td>
</tr>
<tr>
<td>2.20um</td>
<td>0.46</td>
<td>0.42 (-8.7%)</td>
</tr>
<tr>
<td>2.75um</td>
<td>0.38</td>
<td>0.37 (-2.6%)</td>
</tr>
<tr>
<td>3.30um</td>
<td>0.35</td>
<td>0.34 (-2.9%)</td>
</tr>
</tbody>
</table>

- 16-bit bus each a 10mm-long line, 500um per segment
- GISS is up to 39% better than SISS
- ELR-based approach achieves best results and is 100x faster than bottom-up based approach
Flexibility of TRIO

- Different combinations of optimization techniques, e.g.,
  - **T+B+W**: Topology (T), followed by optimal buffer insertion and sizing (B), then followed by optimal wire sizing (W)
  - **TB+BW**: Simultaneous T and B, followed by simultaneous buffer and wire sizing (BW)
  - **TBW**: Simultaneous topology, buffer, and wire optimization

- Different models
  - Simple or table-based model for device delay and interconnect cap
  - Elmore or higher-order delay models

- Different objective functions:
  - Minimize delay under size constraints
  - Minimize power under required arrival time constraints

- **Integrated under an interactive user front-end**
  - Unified input format, data structure and GUI
Example: Trade-off of Run-Times and Solution Quality

- **T+B+W**: Topology (T), followed by optimal buffer insertion and sizing B (B=10) then followed by optimal wire sizing (W=18)
- **TB+BW**: Simultaneous T and B (B=3), followed by simultaneous driver/buffer and wire sizing (BW) with B=40, W=18
- **Tbw+BW**: Simultaneous TBW with small number of B=3 and W=3, then followed by BW as above
- **TBW**: Simultaneous TBW with larger number of B=10 and W=8
### Trade-off of Run-Times and Solution Quality

<table>
<thead>
<tr>
<th></th>
<th>Algorithms</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T+B+W</td>
<td>TB+BW</td>
<td>Tbw+BW</td>
<td>TBW</td>
<td></td>
</tr>
<tr>
<td><strong>5-pin nets</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay (nS)</td>
<td>0.40</td>
<td>0.39</td>
<td>0.35</td>
<td>0.34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.47</td>
<td>0.48</td>
<td>0.38</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.42</td>
<td>0.41</td>
<td>0.36</td>
<td>0.35</td>
<td></td>
</tr>
<tr>
<td>CPU (S)</td>
<td>0.1</td>
<td>0.1</td>
<td>1.4</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td><strong>10-pin nets</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay (nS)</td>
<td>0.42</td>
<td>0.37</td>
<td>0.34</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.56</td>
<td>0.56</td>
<td>0.44</td>
<td>0.44</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.47</td>
<td>0.45</td>
<td>0.38</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td>CPU (S)</td>
<td>0.8</td>
<td>1.0</td>
<td>6.4</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td><strong>20-pin nets</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay (nS)</td>
<td>0.45</td>
<td>0.43</td>
<td>0.38</td>
<td>0.39</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.54</td>
<td>0.48</td>
<td>0.42</td>
<td>0.41</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.46</td>
<td>0.43</td>
<td>0.38</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td>CPU (S)</td>
<td>1.6</td>
<td>4.0</td>
<td>27.6</td>
<td>350</td>
<td></td>
</tr>
</tbody>
</table>

- Tbw+BW achieves “identical” delays as TBW with 10X smaller run-time