MOUNTAIN-MOVER:

an Intuitive Logic Shifting Heuristic for Improving Timing Slack Violating Paths

Xing Wei, Wai-Chung Tang, Yu-Liang Wu
Department of Computer Science and Engineering
The Chinese University of Hong Kong

Cliff Sze, Charles Jay Alpert
IBM Austin Research Center
11400 Burnet Road
Austin TX 78758
Outline

- Our motivation and objectives
- Rewiring technique
- Mountain-mover framework
Why combine logic synthesis and physical design?

- Traditional design flow
  - logic synthesis followed by placement and routing

- No longer good enough for modern designs
  - fail to route
  - fail to meet timing constraints
Logic synthesis in physical design gives another choice to optimize circuits
- Compatible with physical optimization techniques

Initial Circuit
TNS = -77.30

MV
Gain 15.88

PO
Gain 15.42

MV
Gain 10.11

PO
Gain 0.38

PO
Gain 9.2

-61.42

-61.88

-61.88

-61.50

-51.77

-52.22

*PO is a Physical Optimization just trying to move gates in critical paths to a better location
Our motivation and objectives

Rewiring technique

Mountain-mover framework
What is rewiring?

target wire (TW)

alternative wire (AW)
One rewiring operation

- Netlist operation
  - Remove one or more wires (target wire, TW)
  - Add one or more wires (alternative wire, AW)

- Preserve the circuit’s functionality
Why do we use rewiring?

- Powerful
  - about 40% unwanted wires are removable by adding just one alternative wire*

- Efficient
  - 7000 cells in 1000 seconds by functional simulation**
  - 7000 cells in 70 seconds by rewiring

---


- Our motivation and objectives
- Rewiring technique
- Mountain-mover framework
Circuit delay
- The largest signal delay among all paths from any primary inputs (PI) to any primary output (PO).

Timing constraint
- The maximum allowed circuit delay

Arrival time
- The latest time a signal arrives at a pin/gate from the Pis

Require time
- The latest time the data is required to be present at a pin/gate in order to fulfill the timing requirement at the POs
Slack
- The difference (gap) between arrival time and required arrival time.

Worst negative slack (WNS)
- The largest violations among all slacks in the circuit

Total negative slack (TNS)
- $TNS = \sum negative \ slack_i$
Given a logic netlist, its physical placement, and the timing constraints,
the logic restructuring-based placement timing optimization problem is to maximize the worst slack of the circuit by

- (i) re-structuring local circuit while preserving the circuit’s functionality
- (ii) re-placing gates while keeping the placement free of overlaps.
Difficulties for logic restructure

- NP-problem
  - Large solution space
  - Heuristic is necessary
- More complex than physical
  - Fast restructuring techniques
  - Efficient heuristic is necessary
Key idea of mountain-mover I

- Logic rewiring
  - Shift logics to its nearby area

- Slack distribution graph
  - a mapping from the locations of circuit cells to their slack values.
Key idea of mountain-mover II

Initial slack mountain

Optimizing slack mountain from peak first

Optimizing slack mountain from boundary first
Rewire One Wire

1. Given TW & slack distribution
2. Identify Subcircuit
3. Call rewiring engine
   - If No AW, return to step 1
   - If Best AW found, proceed
4. Update Best AW
5. Local evaluation
6. Restore
7. Restructure
   - If Fail, Prune
   - If Prune, For each AW candidate
8. Change netlist
   - If Find best AW, proceed
   - If No best AW, return to step 7
9. Double check
   - If Fail, Restore
   - If Restore, return to step 6
10. No best AW
Prune AW candidates

- Slack gradient: $\vec{g}_t$
- Distance vector: $\vec{d}$
- $\text{slack}_{aw} - \text{slack}_{tw} > 0$
- $\vec{g}_t \cdot \vec{d} > 0$
local worst negative slack (LWNS)
local total negative slack (LTNS)

Given a set of gates $G = \{g_1, g_2, \ldots, g_n\}$,

- $LWNS(G) = \min \{\text{slack}(pt(g_i))\}$
- $LTNS(G) = \sum_{i=1}^{n} \text{slack}(pt(g_i))$

*p_i(g_i) denotes a gate set in which each gate is a primary output and belongs to the fanout cone of a gate g_i*
AW is adopted if

- LWNS is no worse
- Gains the most LTNS improvement
Double check best AW

- Global STA
  - WNS
    - Better
    - No worse
      - TNS
- Roll back if necessary
## Comparison with a recent work

<table>
<thead>
<tr>
<th>Circuit</th>
<th>#cell</th>
<th>#net</th>
<th>Recent work*</th>
<th>Mountain mover (sea upwards)</th>
<th>Peak downwards</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>%impr</td>
<td>time</td>
<td>%impr</td>
<td>time</td>
<td>%impr</td>
</tr>
<tr>
<td>sasc</td>
<td>563</td>
<td>568</td>
<td>14.1</td>
<td>41</td>
<td>44.78</td>
</tr>
<tr>
<td>spi</td>
<td>3227</td>
<td>3277</td>
<td>10.9</td>
<td>949</td>
<td>14.63</td>
</tr>
<tr>
<td>des_area</td>
<td>4881</td>
<td>5122</td>
<td>12.3</td>
<td>503</td>
<td>13.26</td>
</tr>
<tr>
<td>tv80</td>
<td>7161</td>
<td>7179</td>
<td>9.1</td>
<td>1075</td>
<td>8.99</td>
</tr>
<tr>
<td>s35932</td>
<td>7273</td>
<td>7599</td>
<td>27.5</td>
<td>476</td>
<td>16.21</td>
</tr>
<tr>
<td>systemcaes</td>
<td>7959</td>
<td>8220</td>
<td>13.9</td>
<td>748</td>
<td>8.43</td>
</tr>
<tr>
<td>s38417</td>
<td>8278</td>
<td>8309</td>
<td>11.7</td>
<td>481</td>
<td>6.75</td>
</tr>
<tr>
<td>mem_ctrl</td>
<td>11440</td>
<td>11560</td>
<td>9.2</td>
<td>678</td>
<td>18.33</td>
</tr>
<tr>
<td>ac97_ctrl</td>
<td>11855</td>
<td>11948</td>
<td>6.3</td>
<td>245</td>
<td>1.77</td>
</tr>
<tr>
<td>usb_funct</td>
<td>12808</td>
<td>12968</td>
<td>12.2</td>
<td>605</td>
<td>12.92</td>
</tr>
<tr>
<td>DMA</td>
<td>19118</td>
<td>19809</td>
<td>14.5</td>
<td>845</td>
<td>28.95</td>
</tr>
<tr>
<td>aes_core</td>
<td>20795</td>
<td>21055</td>
<td>6.4</td>
<td>603</td>
<td>3.15</td>
</tr>
<tr>
<td>ethernet</td>
<td>46771</td>
<td>46891</td>
<td>3.7</td>
<td>142</td>
<td>4.70</td>
</tr>
<tr>
<td>average</td>
<td>11.7</td>
<td>569</td>
<td>14.14</td>
<td>81.5</td>
<td>9.2</td>
</tr>
<tr>
<td>ratio</td>
<td></td>
<td>7X</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*S. M. Plaza and I. L. Markov, “Optimizing nonmonotonic interconnect using functional simulation and logic restructuring,” TCAD 2008*
Restructure from boundary first is better than that from peak first
- 14.1% vs 9.2%

Our algorithm obtains more delay reduction compared to a previous work
- 14.1% vs 11.7%

Our algorithm is much faster
- Speed up 7 times
The combination between logic synthesis and physical design can gain more improvement.

We propose a novel heuristic that optimizes the critical paths from less critical area first.
Thank You