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

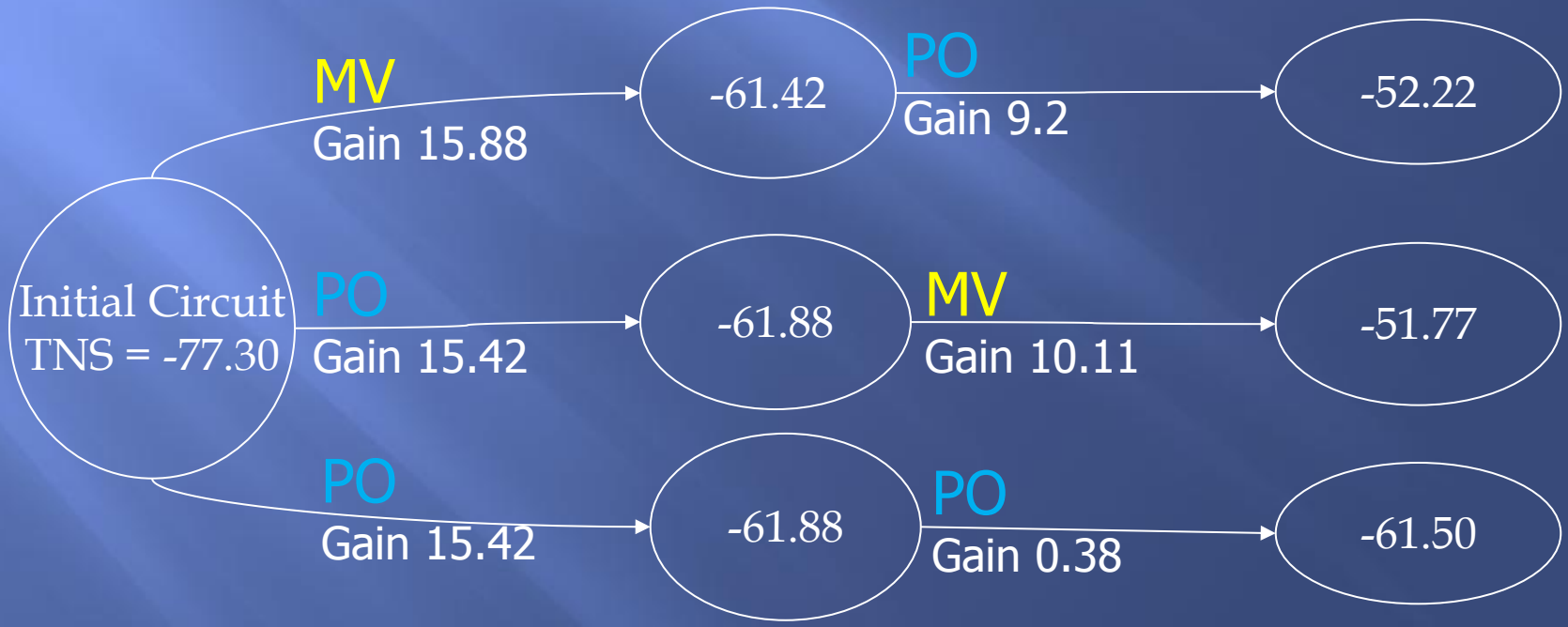
- Our motivation and objectives
- Rewiring technique
- Mountain-mover framework

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

Advantage of the combination

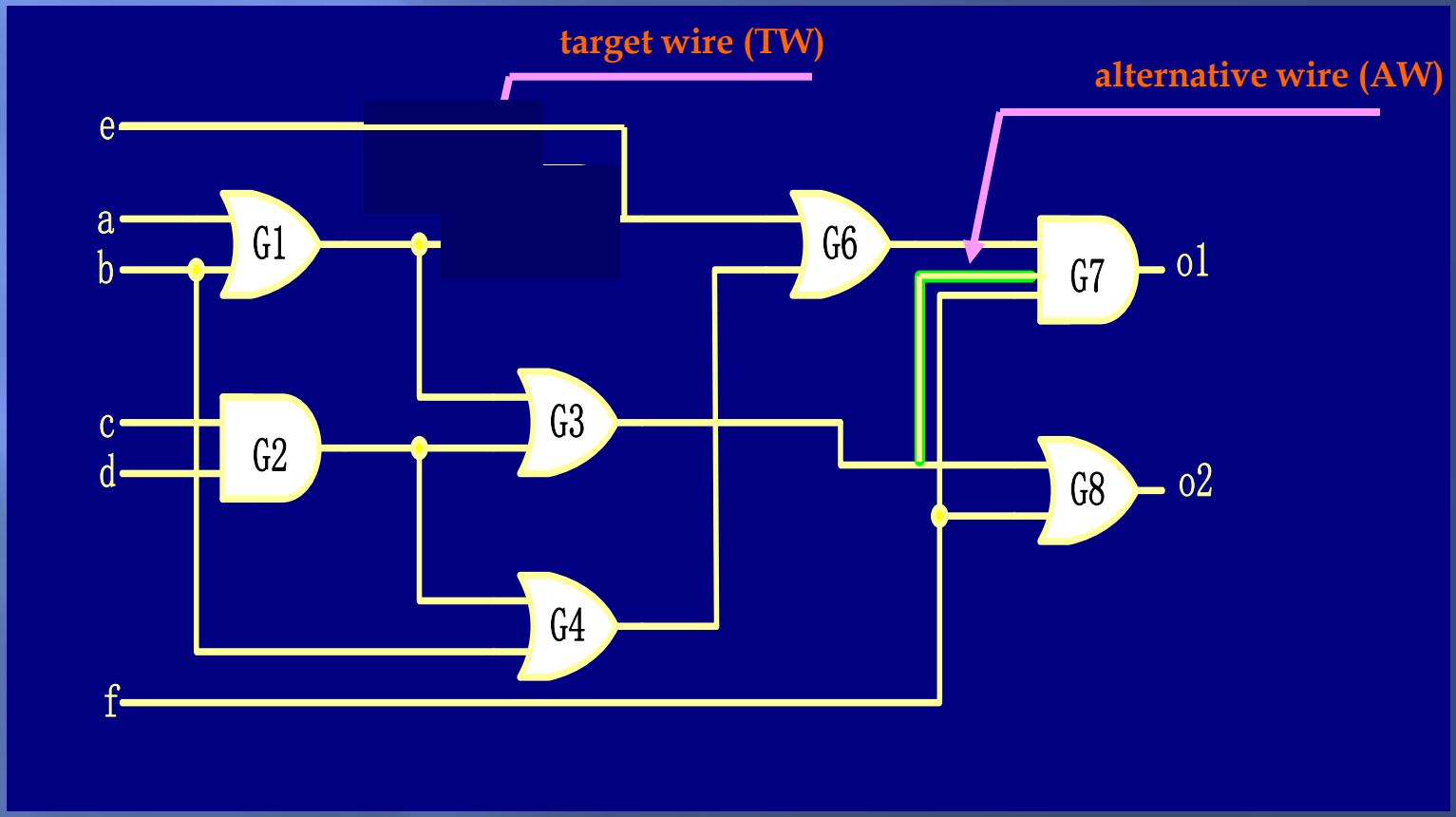
- Logic synthesis in physical design gives another choice to optimize circuits
  - Compatible with physical optimization techniques



*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?



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

**X. Yang, T.-K. Lam, and Y.-L. Wu, "ECR: A low complexity generalized error cancellation rewiring scheme," DAC 2010.*

***S. M. Plaza and I. L. Markov, "Optimizing nonmonotonic interconnect using functional simulation and logic restructuring," TCAD 2008*

- Our motivation and objectives
- Rewiring technique
- **Mountain-mover framework**

Preliminaries

- 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

Preliminaries II

- 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=$\sum negative\ slack_i$

Problem Definition

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

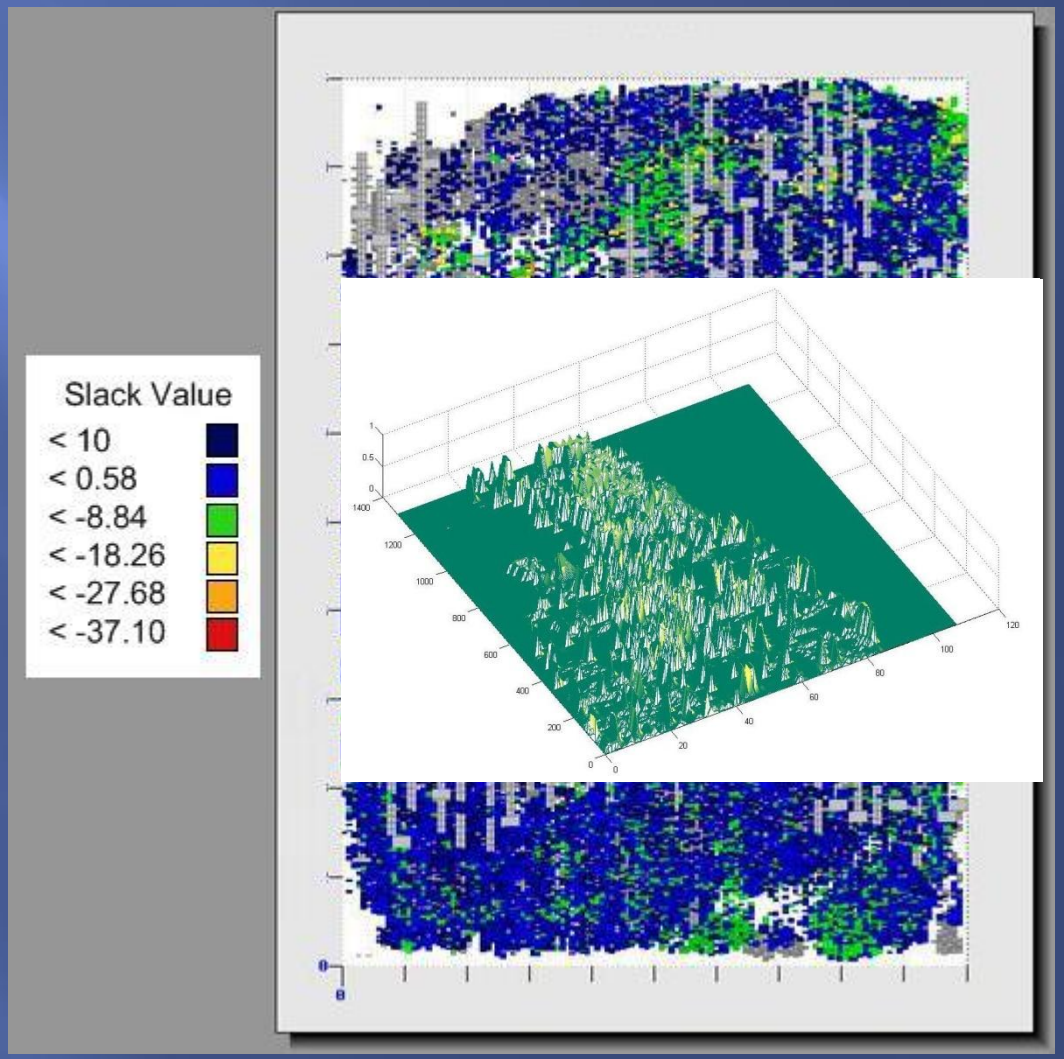
- (i) re-structuring local circuit while preserving the circuit's functionality
- (ii) re-placing gates while keeping the placement free of overlaps.

Difficulties for logic restructure

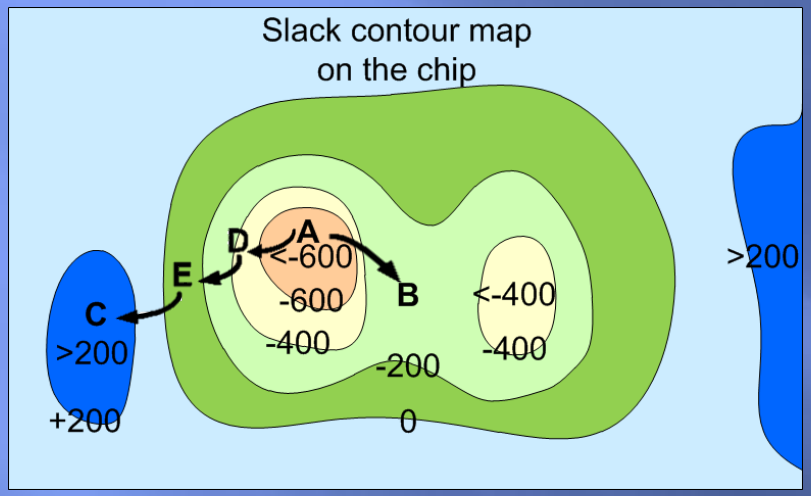
- NP-problem
  - Large solution space
  - Heuristic is necessary
- More complex than physical
  - Fast restructuring techniques
  - Efficient heuristic is necessary

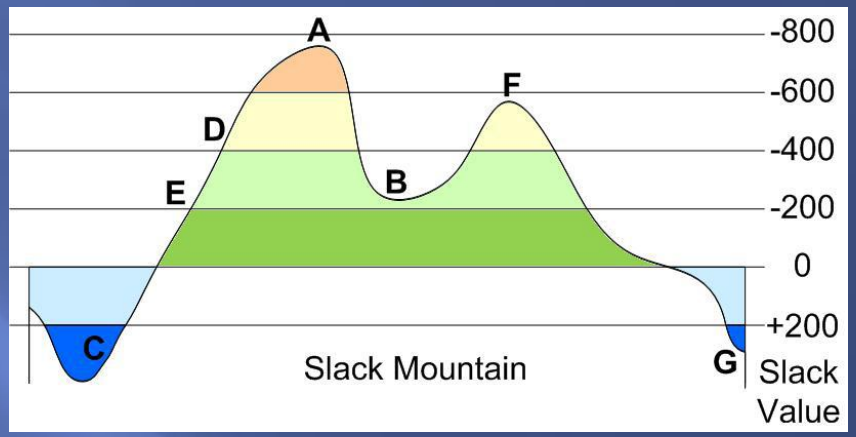
Key idea of mountain-mover I

- Logic rewiring
  - Shift logics to its nearby area
- Slack distribution graph
  - a mapping from the locations of circuit cells to their slack values.

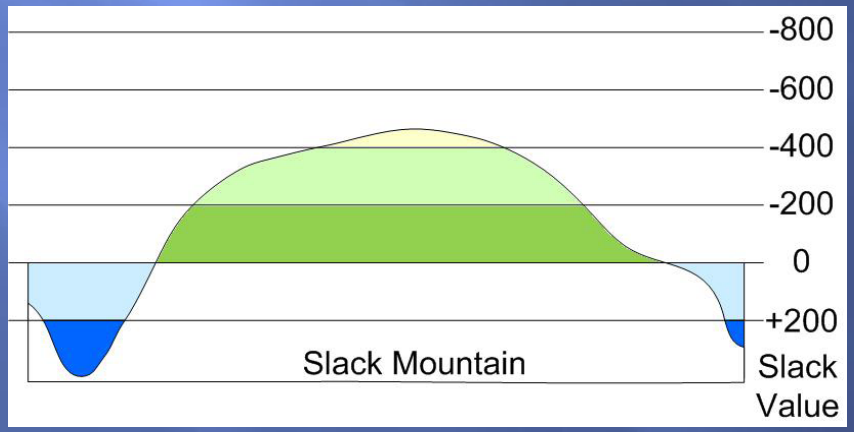


Key idea of mountain-mover II

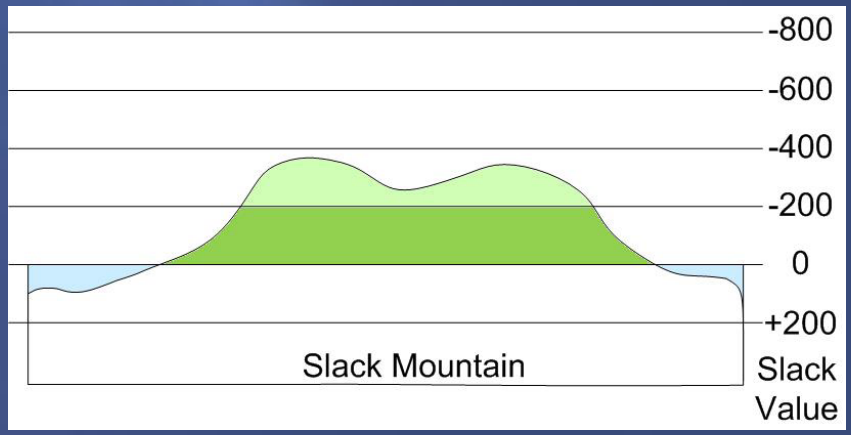




Initial slack mountain

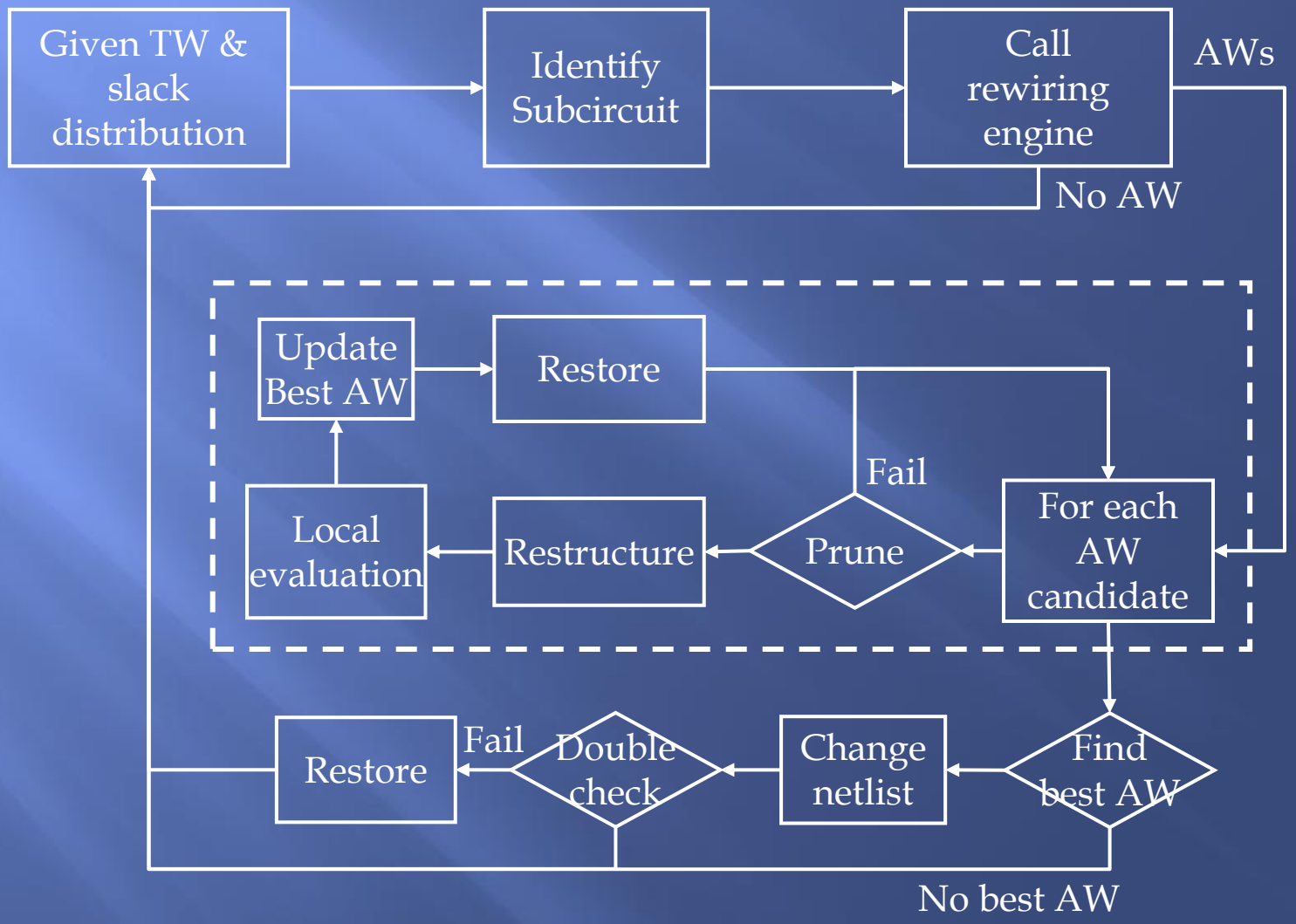


Optimizing slack mountain from peak first



Optimizing slack mountain from boundary first

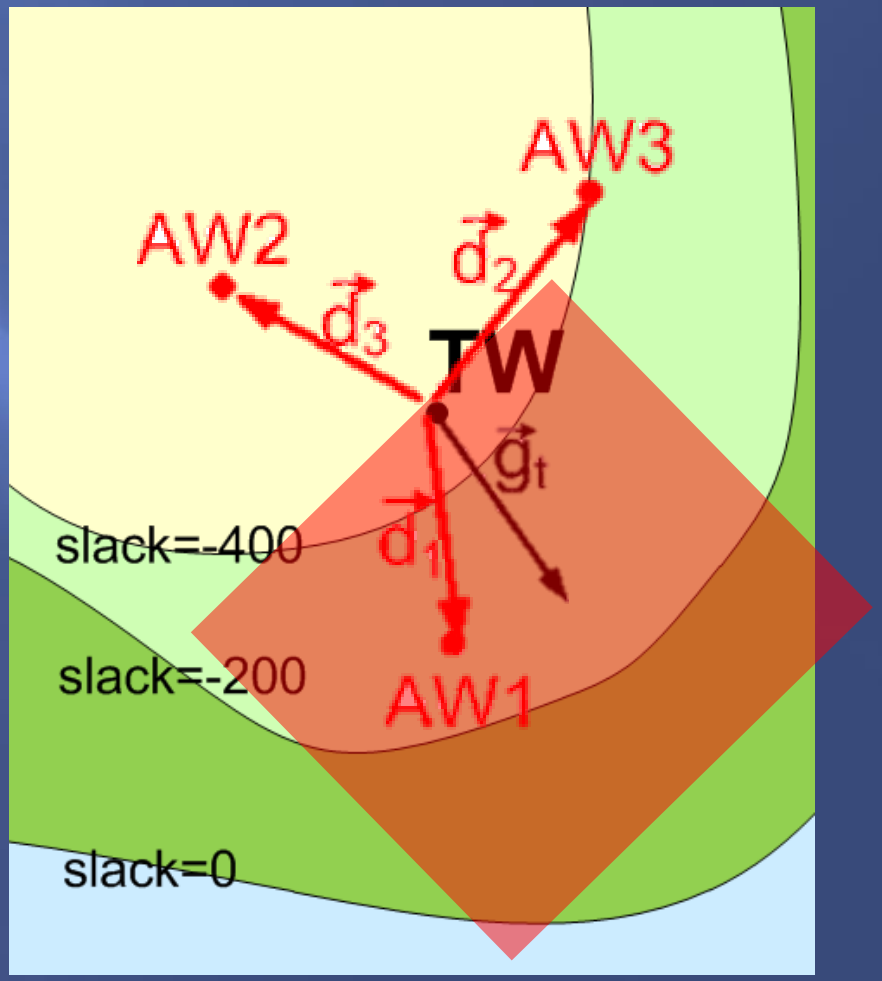
Rewire One Wire



Prune AW candidates

- Slack gradient: $\overrightarrow{g_t}$
- Distance vector: $\vec{d}$

- $slack_{aw} - slack_{tw} > 0$
- $\overrightarrow{g_t} \cdot \vec{d} > 0$



Evaluate AW candidates

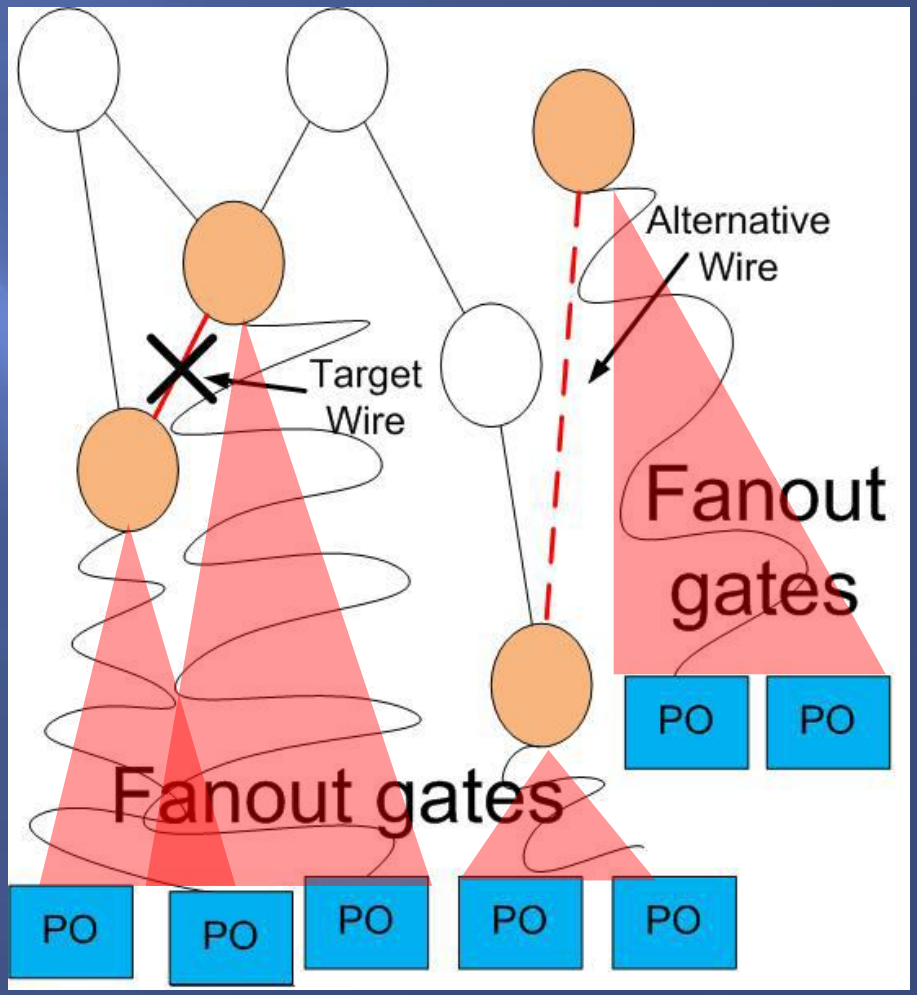
- local worst negative slack (LWNS)
- local total negative slack (LTNS)

- Given a set of gates G = {g_1, g_2, …, g_n},
  - $LWNS(G) = \min\{slack(pt(gi))\}$*
  - $LTNS(G) = \sum_{i=1}^{n} slack(pt(gi))$

*$p_t(g_i)$ denotes a gate set in which each gate is a primary output and belongs to the fanout cone of a gate g_i

Evaluate AW candidates II

- 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

Circuit	#cell	#net	Recent work*		Mountain mover (sea upwards)		Peak downwards
			%impr	time	%impr	time	%impr
sasc	563	568	**14.1**	**41**	**44.78**	**1.6**	**30.0**
spi	3227	3277	**10.9**	**949**	**14.63**	**23.8**	**10.4**
des_area	4881	5122	**12.3**	**503**	**13.26**	**119.3**	**8.1**
tv80	7161	7179	**9.1**	**1075**	**8.99**	**67.1**	**3.4**
s35932	7273	7599	**27.5**	**476**	**16.21**	**19.7**	**14.3**
systemcaes	7959	8220	**13.9**	**748**	**8.43**	**39.6**	**4.0**
s38417	8278	8309	**11.7**	**481**	**6.75**	**38.1**	**6.2**
mem_ctrl	11440	11560	**9.2**	**678**	**18.33**	**391.9**	**9.7**
ac97_ctrl	11855	11948	**6.3**	**245**	**1.77**	**7.5**	**1.8**
usb_funct	12808	12968	**12.2**	**605**	**12.92**	**56.0**	**7.6**
DMA	19118	19809	**14.5**	**845**	**28.95**	**205.4**	**19.8**
aes_core	20795	21055	**6.4**	**603**	**3.15**	**74.1**	**2.4**
ethernet	46771	46891	**3.7**	**142**	**4.70**	**16.0**	**2.2**
average			*11.7*	**569**	*14.14*	**81.5**	*9.2*
ratio				*7X*		*1*	

**S. M. Plaza and I. L. Markov, "Optimizing nonmonotonic interconnect using functional simulation and logic restructuring," TCAD 2008*

Analysis

- 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

Conclusion

- 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