A Unified Framework Processing False Paths and Multi-cycle Paths in Static Timing Analysis

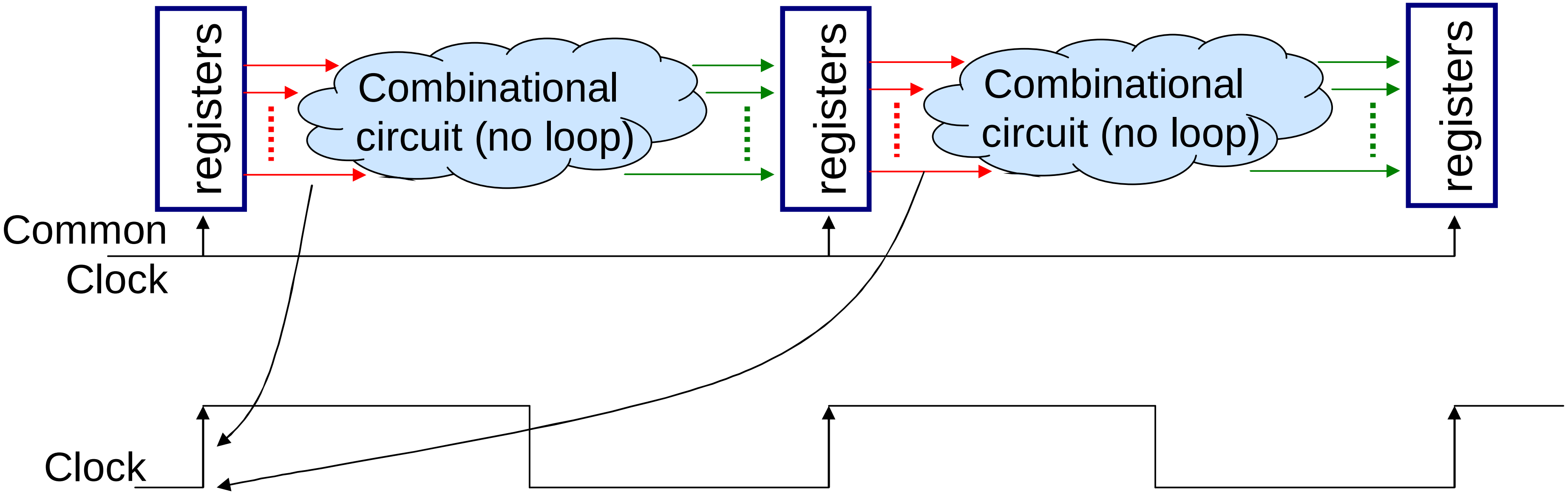
Shuo Zhou, Bo Yao, Hongyu Chen,
Yi Zhu, Chung-Kuan Cheng (UC San Diego),
Mike Hutton (Altera Corp.)

Outline

- Background and Problem Statement
  - Static Timing Analysis (STA)
  - False Paths and Multi-cycle Paths
  - Previous Works on False Paths
- Our Contributions
  - Unified Framework: Rules and Rule Sets
  - Rule Collection Minimization: Extension of our Previous Work [4]
- Experimental Results
- Conclusions

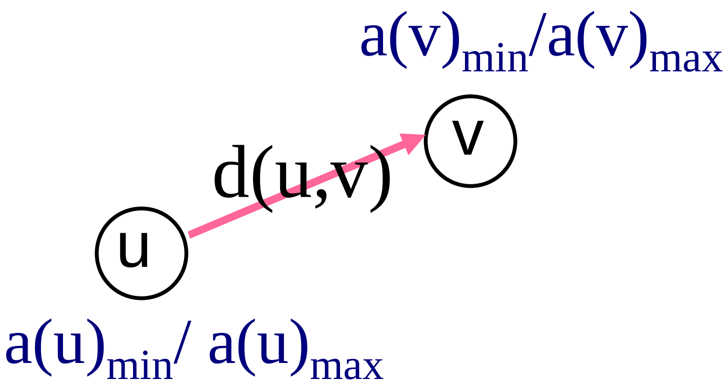
Overview of STA [1]

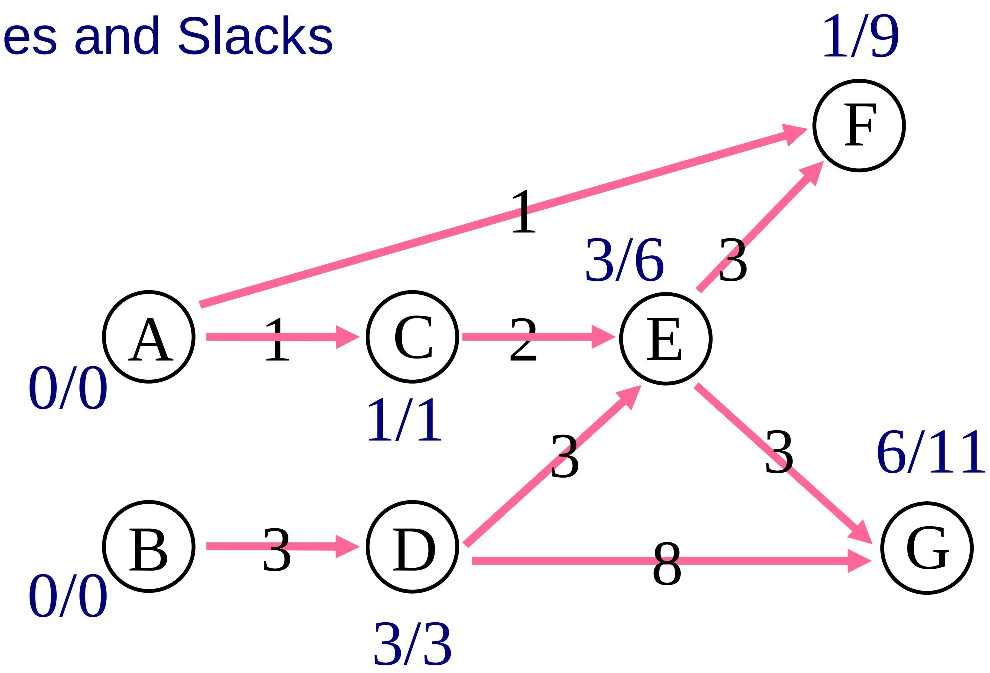
- Formulate combinational circuits between latches into graphs, G = {V, E}



Overview of STA (cont)

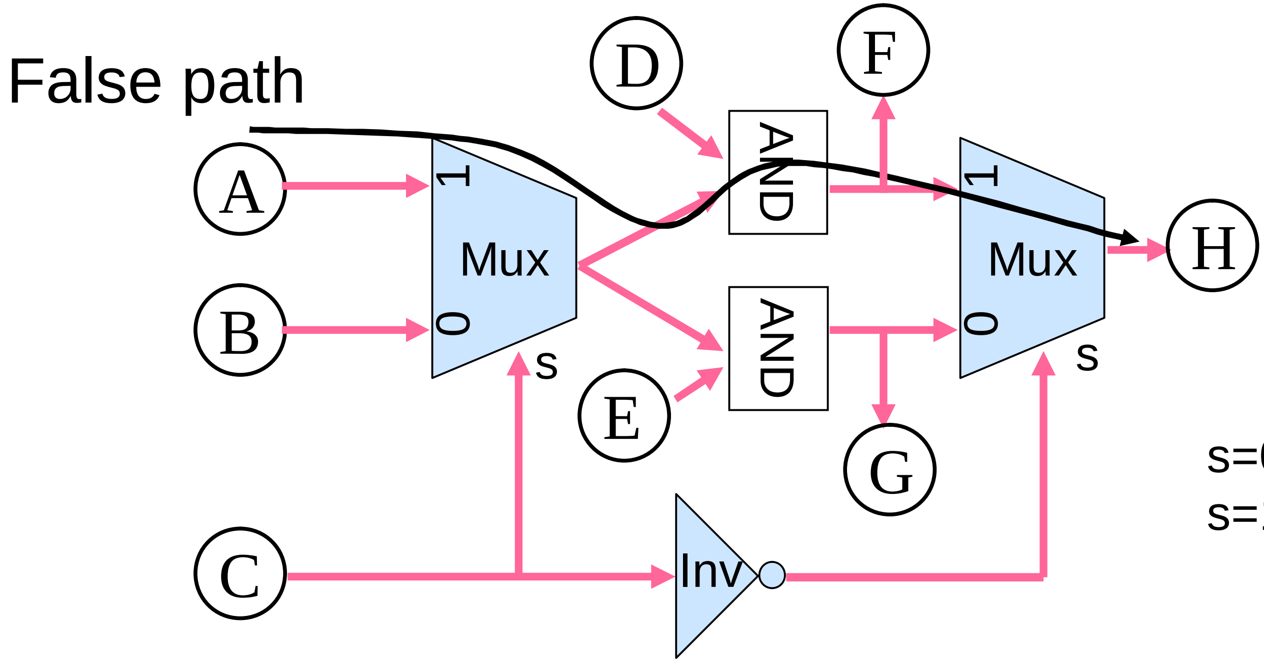
- Dynamic programming
  - One forward sweeping: longest/shortest paths
    - $a(v)_{min}=\min(a(u)_{min}+d(u,v))$, $a(v)_{max}=\max(a(u)_{max} + d(u,v))$
    - General delay bounds, $h \leq delay(path) \leq s$
  - One backward sweeping
    - Required arrival times and Slacks





False Paths

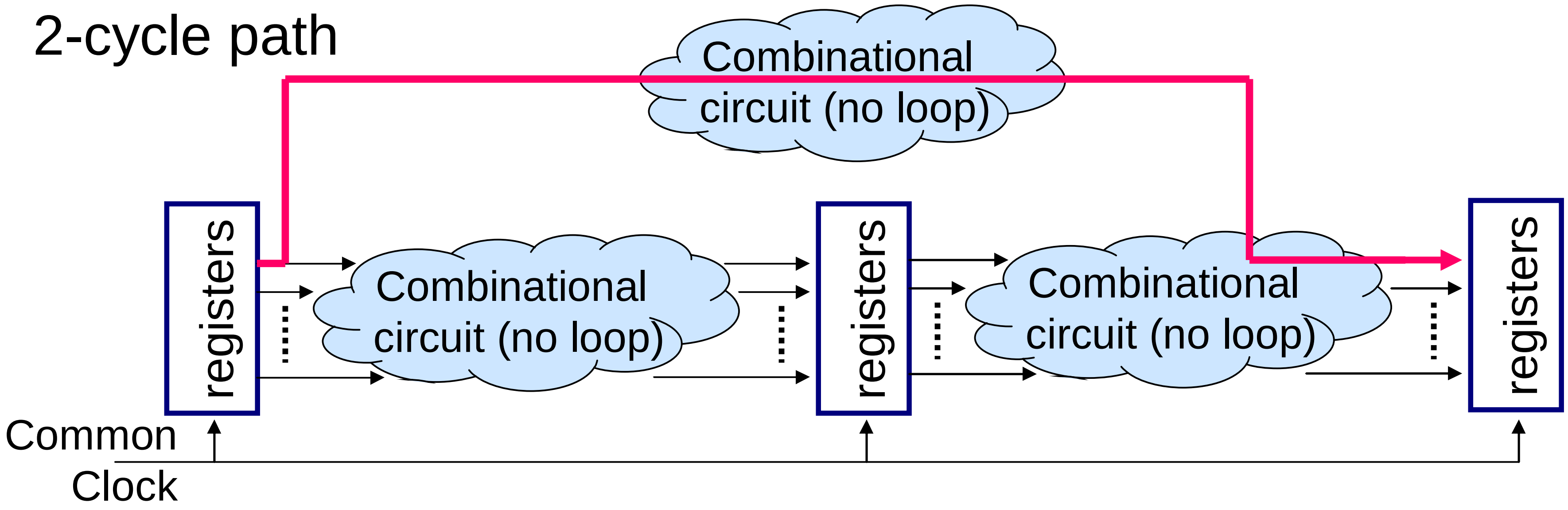
- A path not logically realizable.



s=0, mux.out = mux.input0
s=1, mux.out =mux.input1

Multi-cycle Paths

- A path signals propagate longer than one cycle.



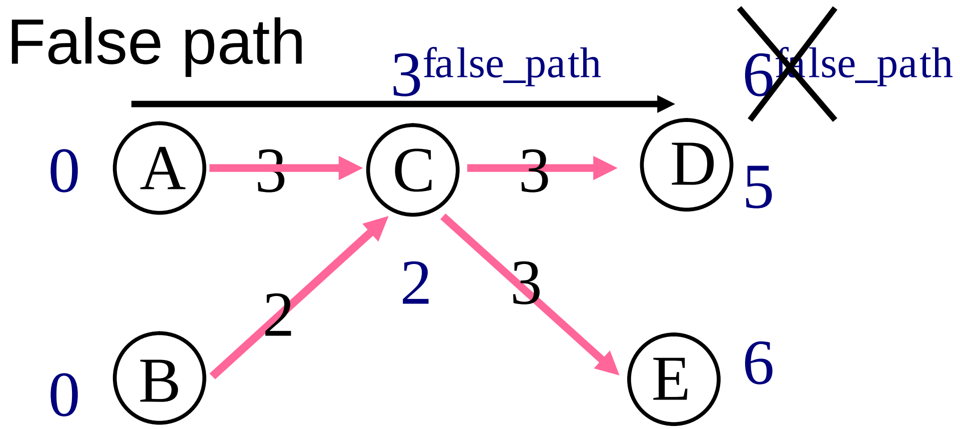
Statement of Problem

Given a set of false paths and multi-cycle paths

- remove false path timing, and compute multi-cycle path slacks with multi-cycle required time in linear time.

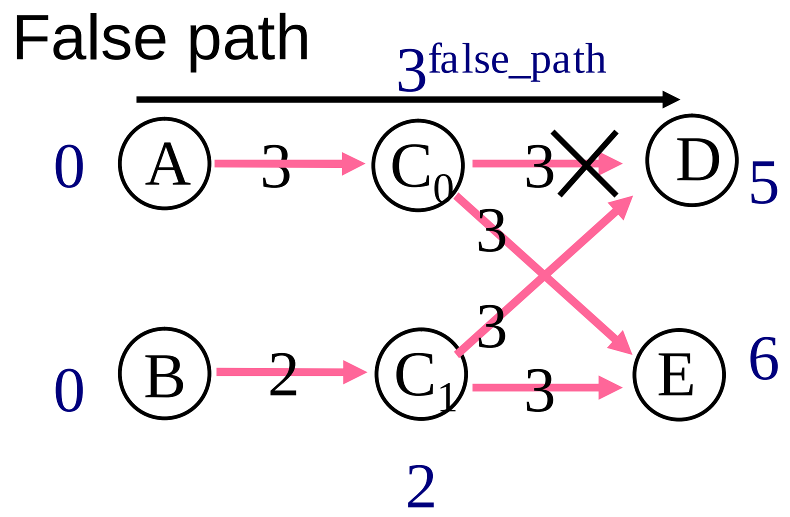
Previous Works on False Paths

- Labeling [2]: remove false path arrival times with tags.



Previous Works (cont)

- Node splitting [3] :
  - Create a new node for each tagged timing
  - Remove false paths by edge removal.
  - Minimize #nodes be split.



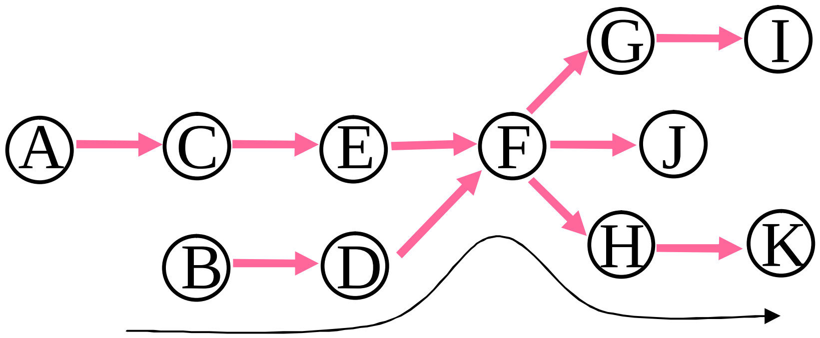
- Two-direction propagation minimizes #tags [4].

Contributions

- Unified Framework
  - Represent false paths and multi-cycle paths as Exceptional Rules.
  - Follow labeling approach[2] to deal with false paths and multi-cycle paths.
- Tag Minimization
  - Follow two-direction propagation[4] to minimize #tags.
  - Minimize #tags = Minimize #distinct arrival times => improve analysis efficiency.

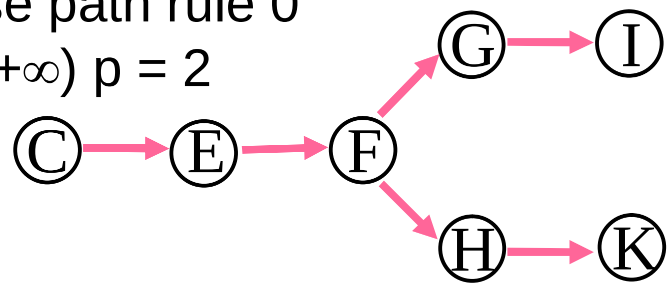
Exceptional Rules

- Rule r
  - $G_r = \{V_r, E_r\}$, sub graph
  - Delay bounds (h_r, s_r), $h_r \leq \text{delay(path)} \leq s_r$.
  - Rule priority p
- Rule sets through an edge, I(u, v)
- Rule sets from and to vertex v, F(v)/T(v)

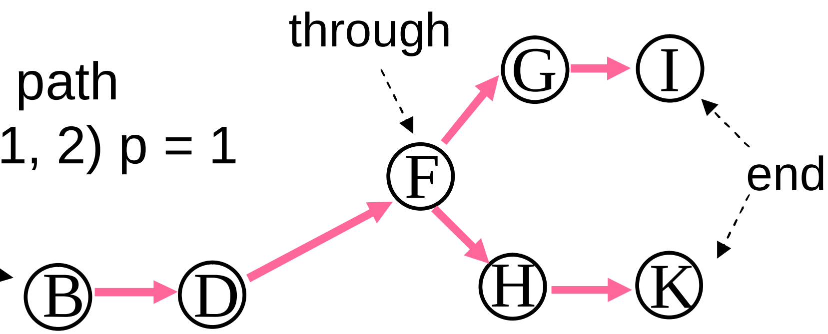


2-cycle path

False path rule 0
$(-\infty, +\infty)$ p = 2

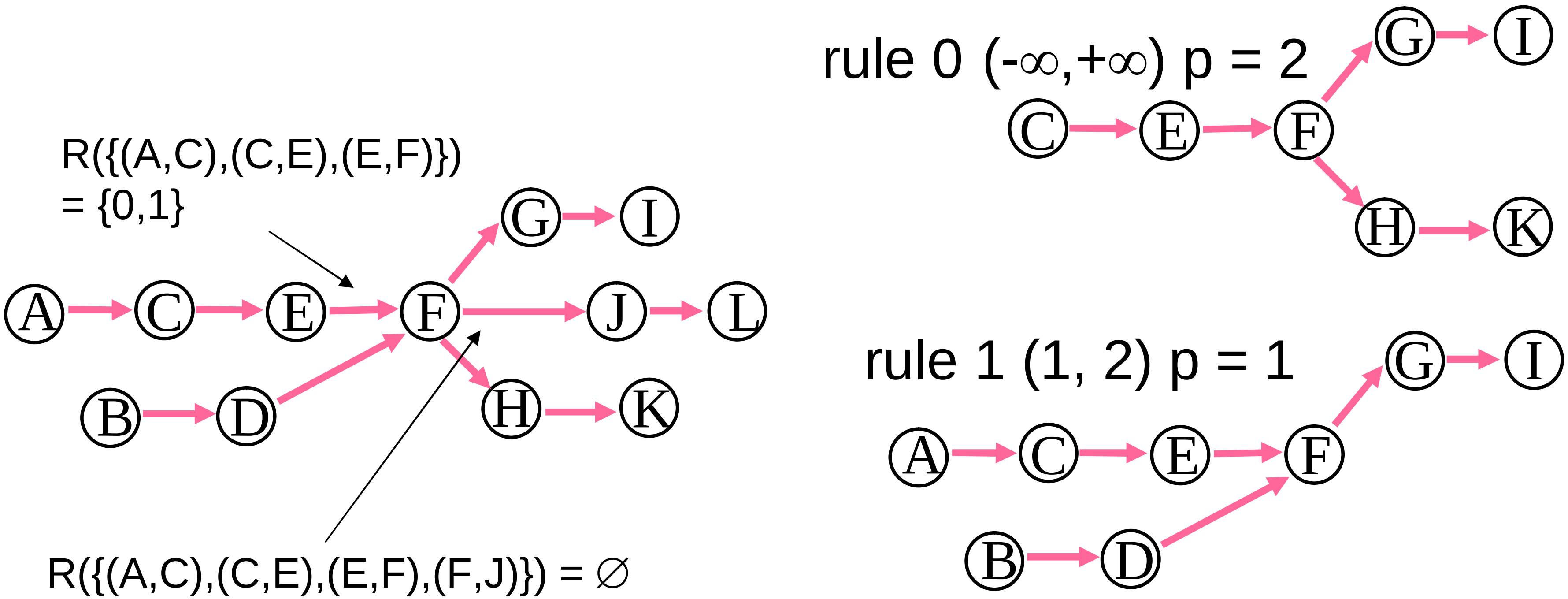


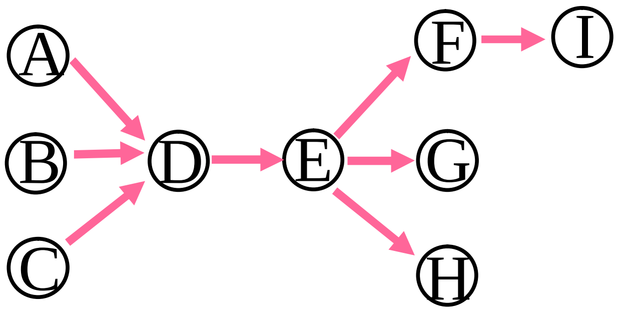
2-cycle path
rule 1 (1, 2) p = 1



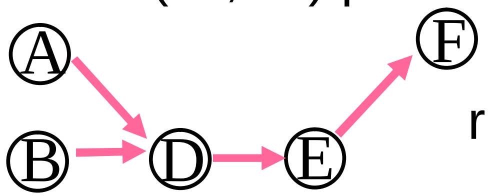
Use Rule Sets as Timing Tags[2]

- Prefix path p^- at vertex v from PI to v
- Prefix Rule Set $R(p^-)$ contains rule r iff p^- belongs to false paths or multi-cycle paths specified by r.

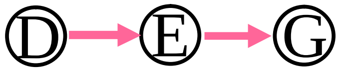




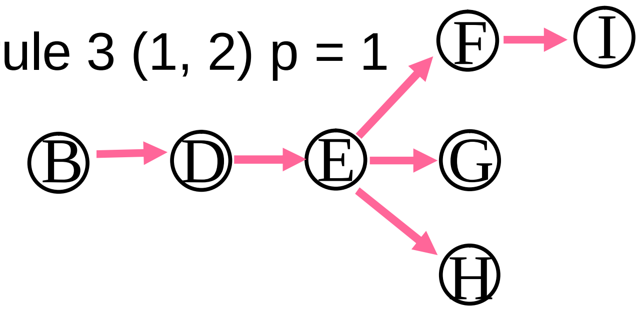
rule 1 (-∞,+∞) p = 2



rule 2 (-∞,+∞) p = 2



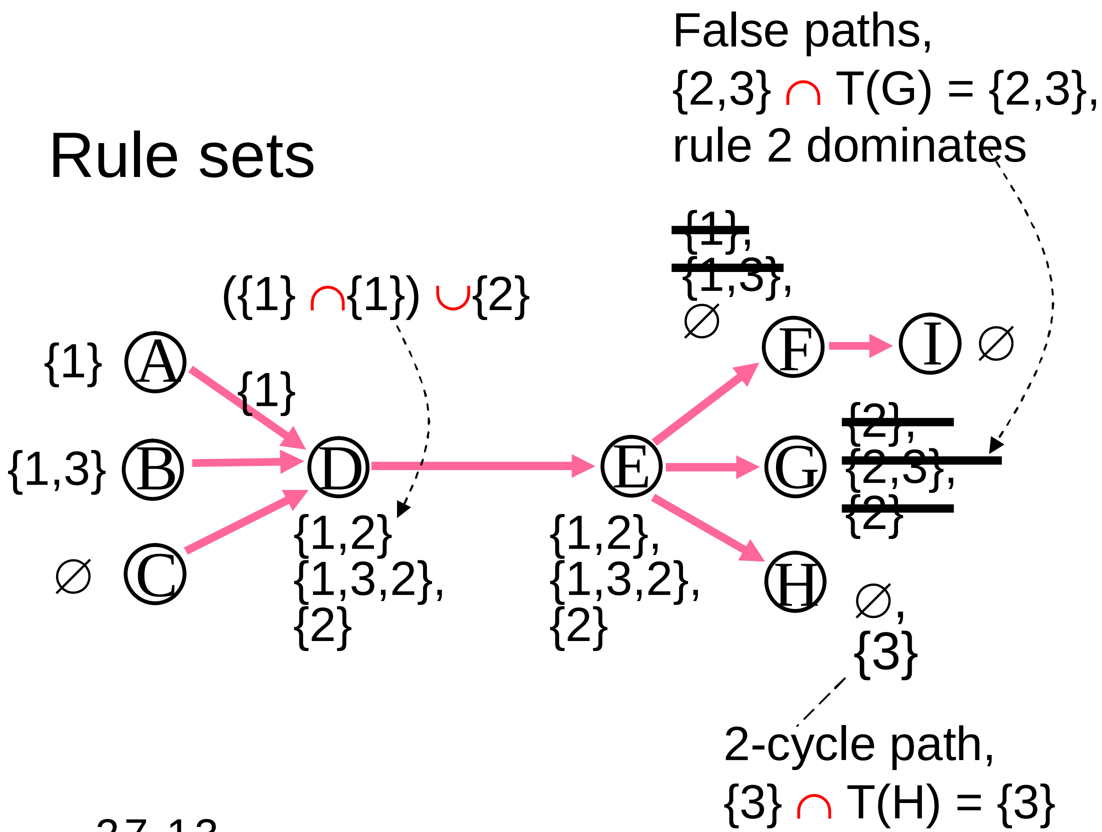
rule 3 (1, 2) p = 1



Rule sets

Compute Rule Sets in Forward Sweeping:

1) Primary inputs => R(p-) = F(v)
2) R(p-)' = (R(p-) ∩I(u,v)) ∪F(v)
3) R(p-)'∩T(v) ≠ ∅ =>
   rule r ∈ R(p-)'∩T(v) with
   highest priority dominates

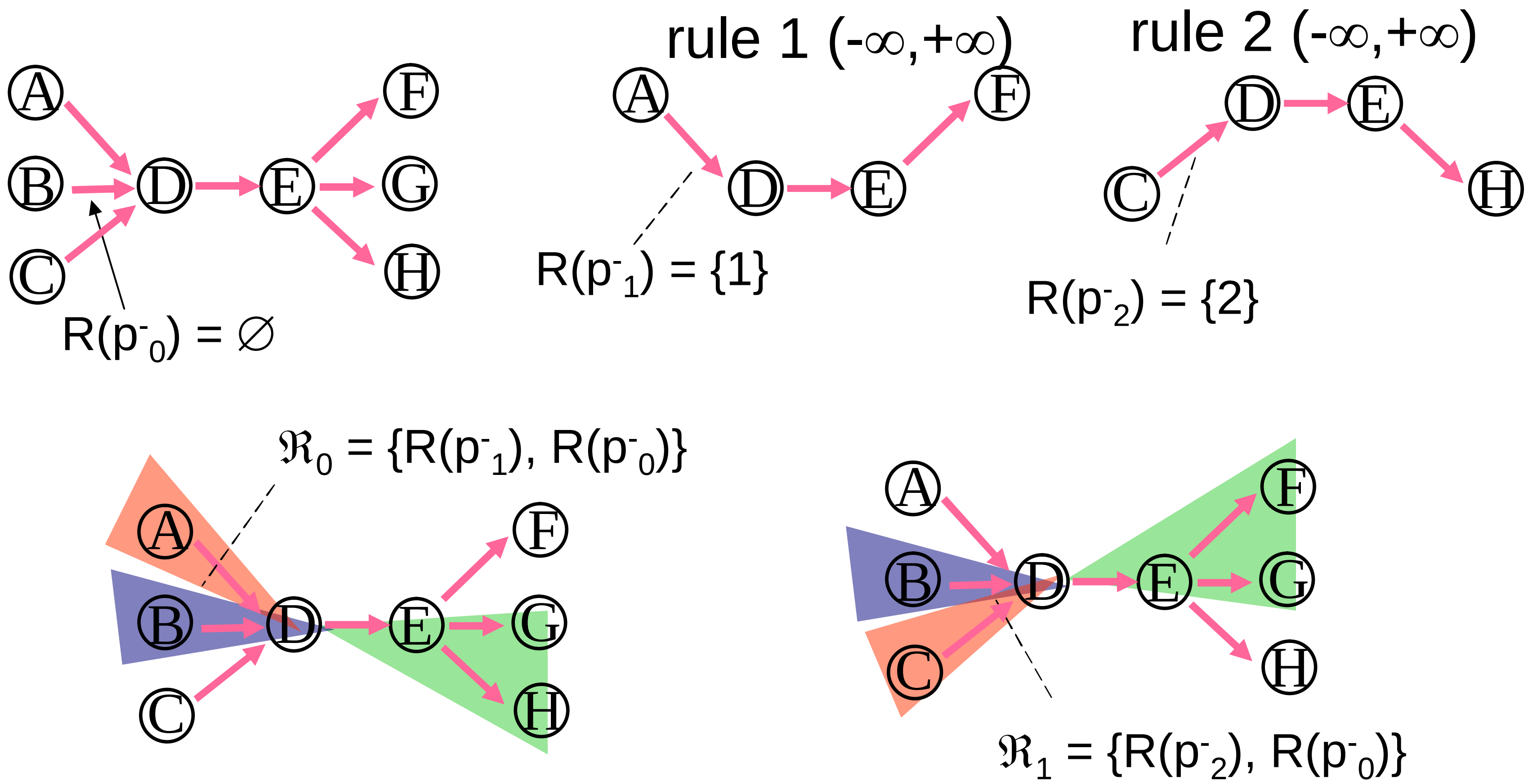


Tag Minimization

- When there are a large number of false paths and multi-cycle paths,
  - ☹Too many tagged timings=>Slow Timing Analysis
- Tag Minimization
  - Follow two-direction propagation [4].
  - Expand to multi-cycle paths.

Basic idea of tag minimization

- Tags can be merged when timing information can be shared.



Rule Collection Minimization

Flow

Backward Sweeping $R(p^+)$

Minimize rule collections at vertex v

Propagate rule collections

Repeat for all vertices in a Forward Sweeping.

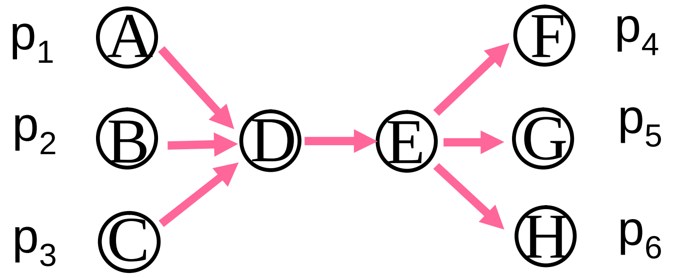
Minimization at vertex v

Construct non-false paths attached with setup times and hold times

Devise timing shifting to align setup times and hold times

Cover distinct non-false path times with rule collections

Intersections and Bipartite Graph

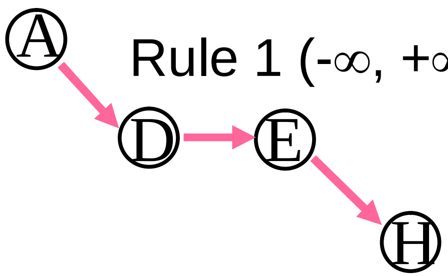


$r \in R(p-) \cap R(p+) \iff$ a path governed by r

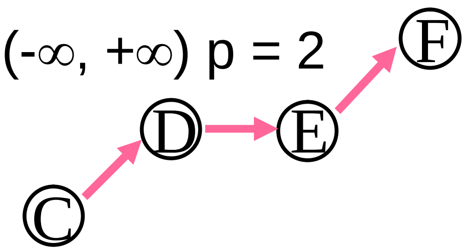
Intersect R(p-) and R(p+) at vertex D

R(p⁻) \ R(p⁺)	$R(p^+_4)=\{2,3\}$	$R(p^+_5)=\{3\}$	$R(p^+_6)=\{1,3\}$
$R(p^-_1)=\{1\}$	∅	∅	~~{1}~~
$R(p^-_2)=\{3\}$	{3}	{3}	{3}
$R(p^-_3)=\{2\}$	~~{2}~~	∅	∅

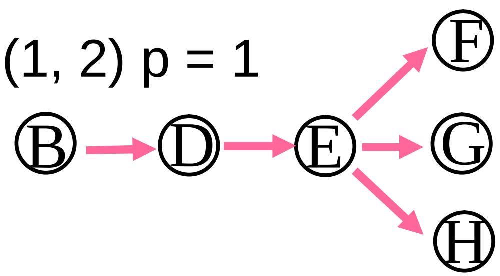
Rule 1 (-∞, +∞) p = 2



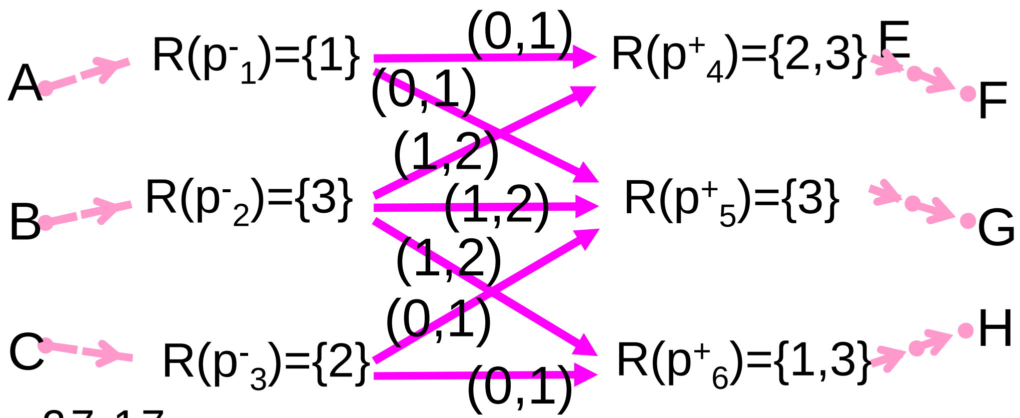
Rule 2 (-∞, +∞) p = 2



Rule 3 (1, 2) p = 1

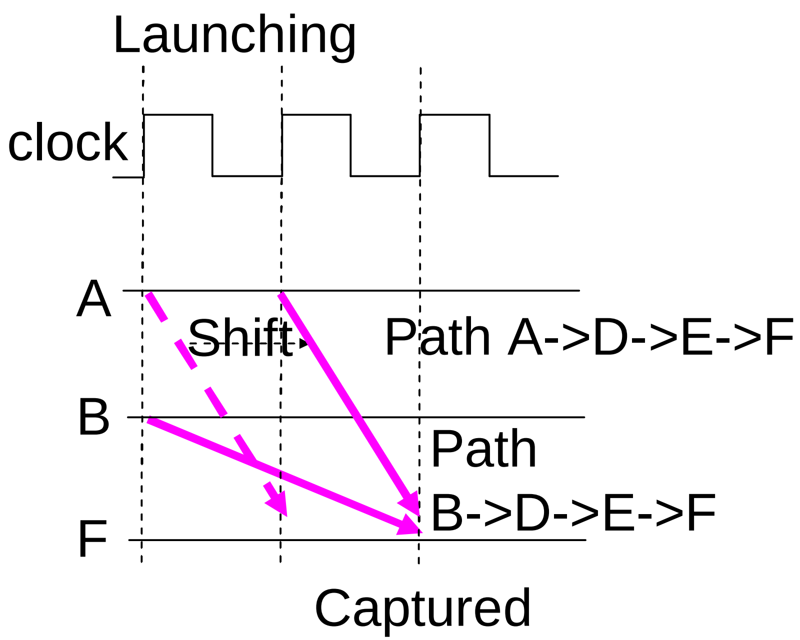


Bipartite graph at D

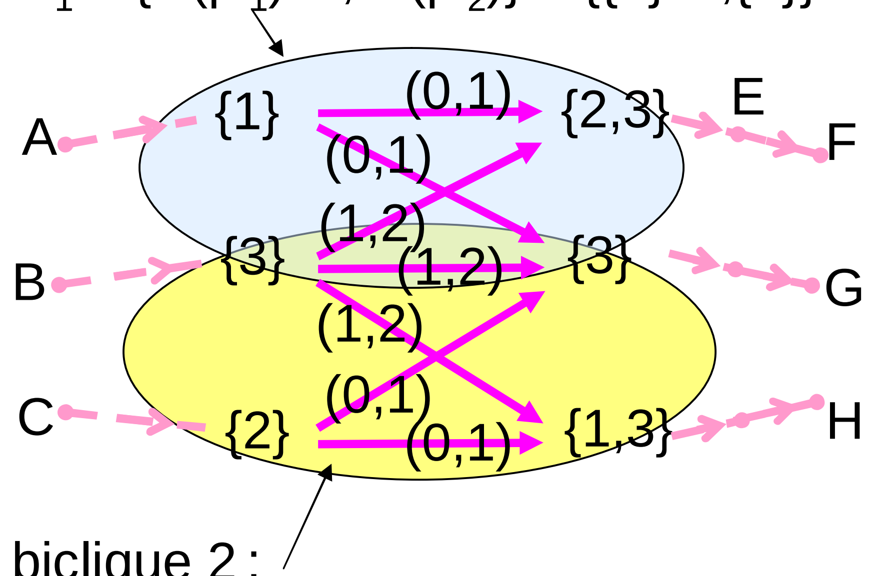


Time Shifting and Biclique Covering [5][6]

Align setup and hold times of two paths.



biclique 1 :
$\Re_1 = \{R(p^-_1)^{+1}, R(p^-_2)\} = \{\{1\}^{+1}, \{3\}\}$



biclique 2 :
$\Re_2 = \{R(p^-_3)^{+1}, R(p^-_2)\} = \{\{2\}^{+1}, \{3\}\}$

$$\Re = \cup \{R(p^-_i)^{+\Delta cycle}\}, R(p^-_i) \in \text{biclique}$$

Propagate Rule Collections

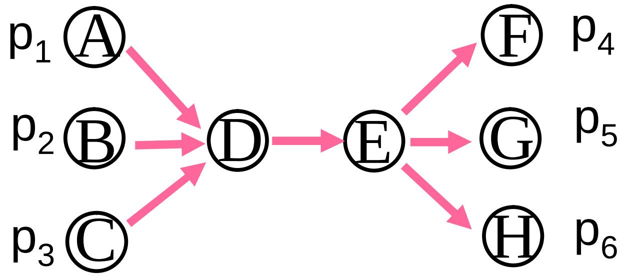
- Propagate every rule set $R(p^-) \in \Re(u)$ from vertex u to v
  - Each $R(p^-)' = (R(p^-) \cap I(u, v)) \cup F(v)$.

D → E

$\Re_1 = \{\{1\}^{+1}, \{3\}\}$ → $\{\{1\}^{+1}, \{3\}\}$

$\Re_2 = \{\{2\}^{+1}, \{3\}\}$ → $\{\{2\}^{+1}, \{3\}\}$

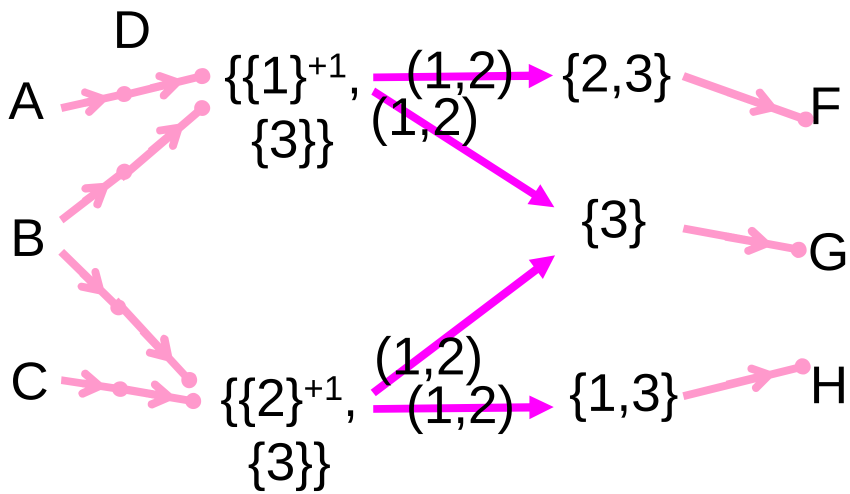
Minimization at Vertex E



Intersections at vertex E

$\mathfrak{R}$ \ $R(p^+)$	$R(p^+_4)=\{2,3\}$	$R(p^+_5)=\{3\}$	$R(p^+_6)=\{1,3\}$
$\mathfrak{R}_1 = \{\{1\}^{+1}, \{3\}\}$	$\{\varnothing^{+1}, \{3\}\}$	$\{\varnothing^{+1}, \{3\}\}$	$\cancel{\{\{1\}^{+1}, \{3\}\}}$
$\mathfrak{R}_2 = \{\{2\}^{+1}, \{3\}\}$	$\cancel{\{\{2\}^{+1}, \{3\}\}}$	$\{\varnothing^{+1}, \{3\}\}$	$\{\varnothing^{+1}, \{3\}\}$

Bipartite graph at E



Setup and hold times Conflict
=> Path Timing is not shared
=> Produce no edge

Correctness

If multi-cycle path rules satisfy:

- sub graph G_r are from primary inputs to primary outputs of graph G

Timing analysis with rule collections as timing tags

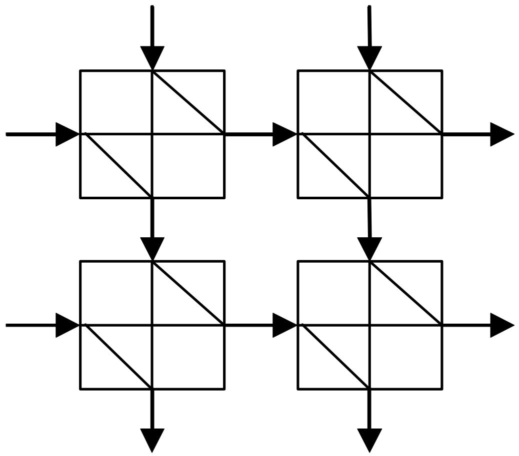
- Computes slacks of non-false paths, and the multi-cycle path slacks are computed using multi-cycle required times.

Complexity

- n: the number of vertices in graph
- k: total number of edges in false paths and multi-cycle paths
- t(v): the number of tags at vertex v is O(k)
- Run time of minimization at v is $O(k^3)$
- Total run time is $O(nk^3)$

Experiments on Artificial Cases

- 100×100 cell mesh.
- Each cell with 2 inputs, and 2 outputs.
- Randomly produced rule.
- Each G_r with 2-4 PIs, 2-4 POs, and 6000 edges.
- $(h_r, s_r) = (1, 2)$ or $(-\infty, +\infty)$



$$\%\text{imp} = \frac{\#\,\text{prefix rule set } R(p^-) - \#\,\text{rule collection } \Re}{\#\,\text{prefix rule set } R(p^-)}$$

Experimental Results

#rules	# Prefix Rule set $R(p^-)$	#Rule collection $\Re(p^-)$	%imp	CPU(s)
9	9129	8281	9.29%	2
34	77102	49321	36.03%	19
69	137581	89987	34.59%	44
88	176384	97124	44.94%	61
104	209718	145484	30.63%	87
average			31.10%	

Experiments on 4 Industry Cases

Case	# nets	# rules		# Prefix Rule Set $R(p^-)$	#Rule Collection $\Re(p^-)$	%imp
		false	Multi-cycle			
tdl	27,555	1	27	9129	8281	9.29%
cq_mod	38,535	2517	3181	77102	49321	36.03%
pm25c	325,582	7	2574	137581	89987	34.59%
atml_ core	533,224	2	2262	176384	97124	44.94%

Run Time Analysis

Case	Minimiza-tion CPU(s)	STA Run Time		
		Use Prefix Rule Set R(p^-)	Use Rule Collection $\Re(p^-)$	%reduction
tdl	2	1.2	1.2	0
cq_mod	19	4.2	2	52.38%
pm25c	44	55.33	28.5	48.49%
atml_core	61	40.33	19.5	51.65%
average				38.13%

$$\%\text{reduction} = \frac{\text{STA runtime using } R(p^-) - \text{STA runtime using } \Re}{\text{STA runtime using } R(p^-)}$$

Conclusions

- We propose a framework to unify the false paths and multi-cycle path constraints .
- We use two-direction propagation approach to produce minimized number of tags for false path and multi-cycle paths. .
- The experimental results on 4 industry test cases show that STA run time is reduced by 38.13% in average. The runtime of the minimization is only 61 seconds for the largest case.

References

[1] R. B. Hitchcock, "Timing Verification and Timing Analysis Program", DAC 1982, pp. 594-604.

[2] K. P Belkhale, et. al., "Timing Analysis with known False Sub-Graphs", ICCAD 1995, pp. 736-740.

[3] D. Blaauw, et. al., "Removing user-specified false paths from timing graphs", DAC 2000, pp. 270-273.

[4] S. Zhou, B. Yao, H. Chen, Y. Zhu, C.K. Cheng, M. Hutton, et al., "Improving the efficiency of Static Timing Analysis with False Paths", IEEE/ACM Int. Conf. CAD 2005, pp. 527-531.

[5] J. Orlin, "Containment in graph theory: Covering graphs with cliques", Nederl. Akad. Wetensch. Indag. Math., 39:211-218, 1977.

[6] H. Muller, "Alternate Cycle-Free Matchings", Order, (7):11-21, 1990.

Thank you!