Statistical Timing Verification for Transparently Latched Circuits through Structural Graph Traversal

Xingliang Yuan and Jia Wang,
Electrical and Computer Engineering Department,
Illinois Institute of Technology, USA

January, 2010

Outline

Timing Verification for Transparently Latched Circuits under Process Variation

- Level-triggered transparent latches: low overhead in timing, area and power.
- Circuit timing under process variation: addressed by many researches using statistical static timing analysis(SSTA).
- Most previous works: assume their timing graphs are acyclic (effective for circuits using flip-flops).
- Recent works on latch timing verification:
  - Chen's *PCycle*[Chen el.at'06] is optimistic for missing cycles through a structural graph traversal.
  - Zhang's work[Zhang el.at'06] convergence depends on the accuracy of the statistical operations and the number of the iterations cannot be bounded.

Timing Verification for Transparently Latched Circuits under Process Variation

- Level-triggered transparent latches: low overhead in timing, area and power.
- Circuit timing under process variation: addressed by many researches using statistical static timing analysis(SSTA).
- Most previous works: assume their timing graphs are acyclic (effective for circuits using flip-flops).
- Recent works on latch timing verification:
  - Chen's *PCycle*[Chen el.at'06] is optimistic for missing cycles through a structural graph traversal.
  - Zhang's work[Zhang el.at'06] convergence depends on the accuracy of the statistical operations and the number of the iterations cannot be bounded.

Timing Verification for Transparently Latched Circuits under Process Variation

- Level-triggered transparent latches: low overhead in timing, area and power.
- Circuit timing under process variation: addressed by many researches using statistical static timing analysis(SSTA).
- Most previous works: assume their timing graphs are acyclic (effective for circuits using flip-flops).
- Recent works on latch timing verification:
  - Chen's *PCycle*[Chen el.at'06] is optimistic for missing cycles through a structural graph traversal.
  - Zhang's work[Zhang el.at'06] convergence depends on the accuracy of the statistical operations and the number of the iterations cannot be bounded.

Timing Verification for Transparently Latched Circuits under Process Variation

- Level-triggered transparent latches: low overhead in timing, area and power.
- Circuit timing under process variation: addressed by many researches using statistical static timing analysis(SSTA).
- Most previous works: assume their timing graphs are acyclic (effective for circuits using flip-flops).
- Recent works on latch timing verification:
  - Chen's *PCycle*[Chen el.at'06] is optimistic for missing cycles through a structural graph traversal.
  - Zhang's work[Zhang el.at'06] convergence depends on the accuracy of the statistical operations and the number of the iterations cannot be bounded.

Our Contribution

- Structural Graph Traversal–Positive Cycle Detection :
  - Covers *all* cycles through a structural graph traversal.
  - Terminates within a *polynomial* number of statistical operation.

Statistical Static Timing Analysis

- Assume the random delay d of a gate or an interconnect is gaussian distributed.
- The **sum** of two gaussian random valuables is still gaussian distributed.
- The **max** of them is approximately gaussian distributed [Clark'62].
- Block based statistical static timing analysis(SSTA)[Chang el.at'04][Visweswariah el.at'04]: uses statistical **max** and **sum** operation to calculate the distribution of maximal path delay from inputs to outputs.

Statistical Static Timing Analysis

- Assume the random delay d of a gate or an interconnect is gaussian distributed.
- The **sum** of two gaussian random valuables is still gaussian distributed.
- The **max** of them is approximately gaussian distributed [Clark'62].
- Block based statistical static timing analysis(SSTA)[Chang el.at'04][Visweswariah el.at'04]: uses statistical **max** and **sum** operation to calculate the distribution of maximal path delay from inputs to outputs.

Statistical Static Timing Analysis

- Assume the random delay d of a gate or an interconnect is gaussian distributed.
- The **sum** of two gaussian random valuables is still gaussian distributed.
- The **max** of them is approximately gaussian distributed [Clark'62].
- Block based statistical static timing analysis(SSTA)[Chang el.at'04][Visweswariah el.at'04]: uses statistical **max** and **sum** operation to calculate the distribution of maximal path delay from inputs to outputs.

Statistical Static Timing Analysis

- Assume the random delay d of a gate or an interconnect is gaussian distributed.
- The **sum** of two gaussian random valuables is still gaussian distributed.
- The **max** of them is approximately gaussian distributed [Clark'62].
- Block based statistical static timing analysis(SSTA)[Chang el.at'04][Visweswariah el.at'04]: uses statistical **max** and **sum** operation to calculate the distribution of maximal path delay from inputs to outputs.

Clock Methodology For Latched Circuits

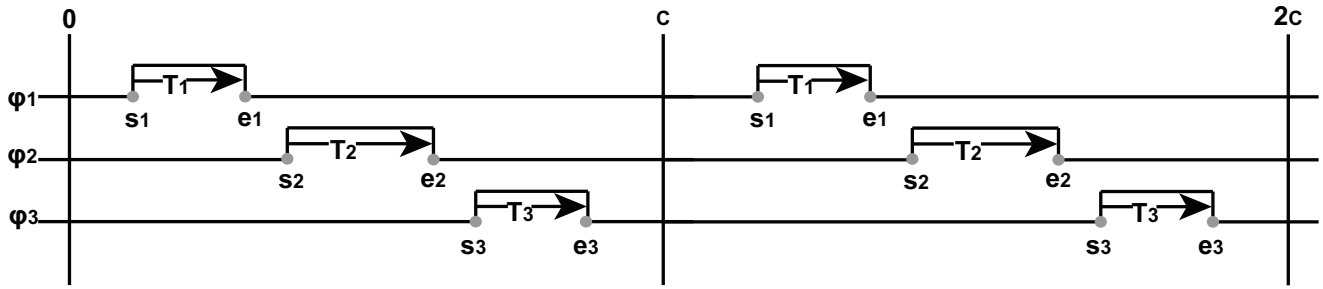


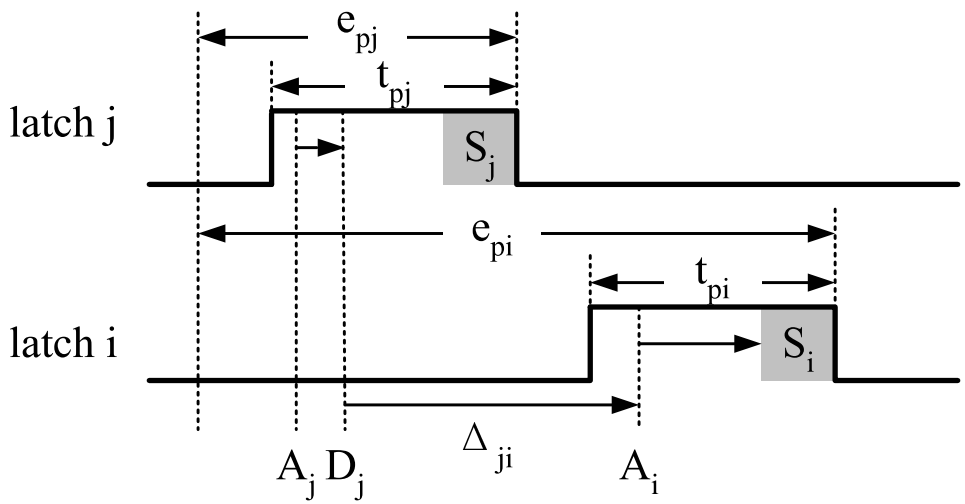
Figure: Three phase non-overlapping clock

- L: the set of the transparent latches in a circuit.
- A clock schedule is the assignment of a clock signal φ_i to each latch $i \in L$.
  - All clock signals should be of the same clock period c.
  - But can have different phases.
- Active interval length T_i is denoted by its starting and ending time (s_i, e_i): $T_i = e_i - s_i$.

Latch Latest Timing for Setup Constraints

Known:
Δ_{ij}: longest combinational path delay from latch i to latch j.
S_i: the setup time of latch i.
Unknown:
A_i: the latest signal arrival time of latch i.
D_i: the latest signal departure time of latch i.

Latest timing constraints[Sakallah et.al'92]:
Max-delay constraint:
$A_i - D_j \geq \Delta_{ji} - E_{p_j p_i}, \ \forall j \rightarrow i,$
Departure time constraints:
$D_j \geq A_j, \ D_j \geq c - t_{p_j}, \ \forall j \in L.$
Setup time constraint:
$A_j \leq c - S_j, \ \forall j \in L.$



(A_i, D_i) are in latch i's local time, i.e. starting at the falling edge of previous clock signal.

Latch Latest Constraints Graph

[Chen el.at'06]

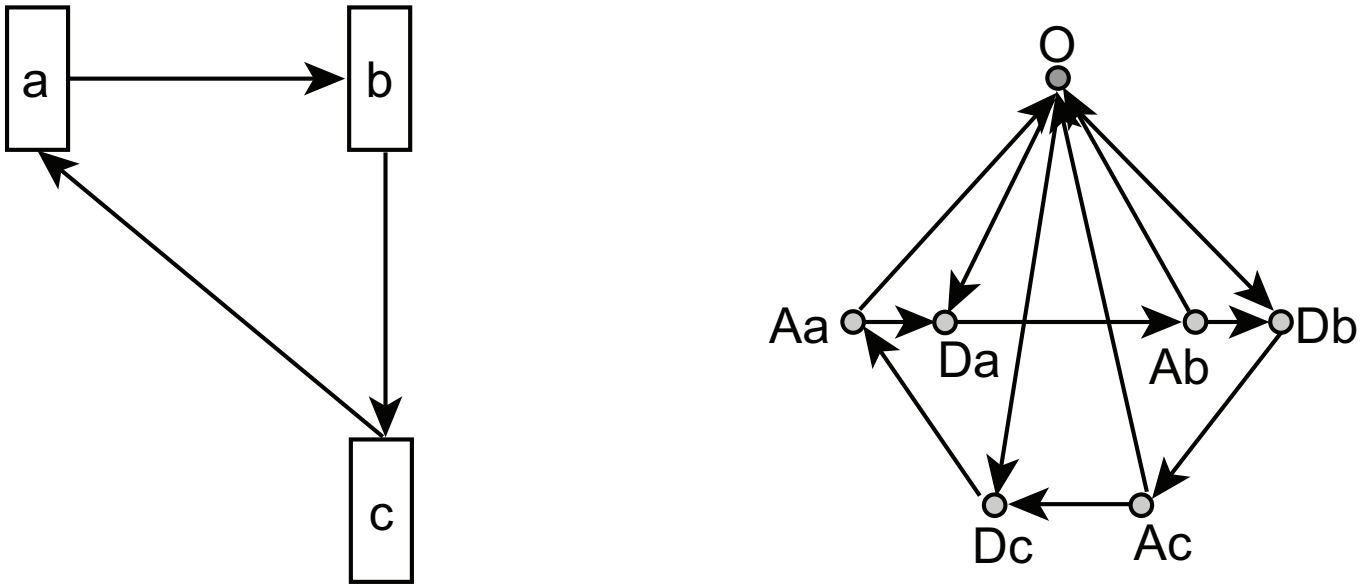


Figure: A three latch circuit and its latest timing constraint graph

Consider a to b, latest timing constraints:

$$A_b - D_a \geq \Delta_{ab} - E_{p_a p_b},$$
$$D_b - A_b \geq 0,$$
$$D_b - O \geq c - t_{p_b},$$
$$O - A_b \geq S_b - c,$$

- Earliest timing constraints graph are constructed similarly for the hold time constraint verification.

Structural Verification of Clock Schedule

- Deterministic situation[Chen el.at'06]: the given clock of a latched circuit is valid iff
  - The latest constraint graph has no positive cycle,
  - The earliest constraint graph has no negative cycle.
- Statistical situation[Chen el.at'06]: the timing yield of the circuit, that the clock schedule is valid, is equal to the **probability** that
  - The latest constraint graph has no positive cycle,
  - The earliest constraint graph has no negative cycle.

Outline

Problem Formulation

Problem (Statistical Positive Cycle Detection)

Let $G = (V, E)$ be a graph. For each edge $(i, j) \in E$, let $w(i, j)$ be the random edge weight. Assume that the joint distribution of w is known. Determine a random variable X and its distribution such that G has a positive cycle iff $X > 0$.

Outline

Algorithmic Idea

- $w(c)$:the cycle weight.

$$w(c) \triangleq \sum_{(i,j)\in c} w(i,j).$$

- X_{all}: the maximum of all the cycle weights.

$$X_{all} \triangleq \max_{\text{cycle } c} w(c).$$

- X_{all} is hard to compute since the number of cycle is infinite in a graph.

Algorithmic Idea

- $w(c)$:the cycle weight.

$$w(c) \triangleq \sum_{(i,j)\in c} w(i,j).$$

- X_{all}: the maximum of all the cycle weights.

$$X_{all} \triangleq \max_{\text{cycle } c} w(c).$$

- X_{all} is hard to compute since the number of cycle is infinite in a graph.

Algorithmic Idea

- $w(c)$:the cycle weight.

$$w(c) \triangleq \sum_{(i,j)\in c} w(i,j).$$

- X_{all}: the maximum of all the cycle weights.

$$X_{all} \triangleq \max_{\text{cycle } c} w(c).$$

- X_{all} is hard to compute since the number of cycle is infinite in a graph.

Algorithmic Idea

- Simple cycle:traverses any vertex at most once.
  - S: the set of simple cycles in G.
  - S: a finite set.

Lemma

Define

$$X_S \triangleq \max_{c \in S} w(c).$$

Then $X_S > 0$ *iff* $X_{all} > 0$.

  - Hard to enumerate simple cycles explicitly.

Lemma

If $S \subseteq S^*$, *then* $X_{S^*} > 0$ *iff* $X_S > 0$.

Algorithmic Idea

- Simple cycle:traverses any vertex at most once.
  - S: the set of simple cycles in G.
  - S: a finite set.

Lemma

Define

$$X_S \triangleq \max_{c \in S} w(c).$$

Then $X_S > 0$ *iff* $X_{all} > 0$.

  - Hard to enumerate simple cycles explicitly.

Lemma

If $S \subseteq S^*$, *then* $X_{S^*} > 0$ *iff* $X_S > 0$.

Algorithmic Idea

- Simple cycle:traverses any vertex at most once.
  - S: the set of simple cycles in G.
  - S: a finite set.

Lemma

Define

$$X_S \triangleq \max_{c \in S} w(c).$$

Then $X_S > 0$ *iff* $X_{all} > 0$.

- Hard to enumerate simple cycles explicitly.

Lemma

If $S \subseteq S^*$, *then* $X_{S^*} > 0$ *iff* $X_S > 0$.

SGT-PC Algorithm

- L_i: the set of cycles in G that contain at most $|V|$ vertices and traverse the vertex i exactly once.

$$S \subseteq L \triangleq \bigcup_{i \in V} L_i.$$

- Defining $X_i \triangleq \max_{c \in L_i} w(c)$, we have,

$$X_L = \max_{i \in V} X_i.$$

SGT-PC Algorithm

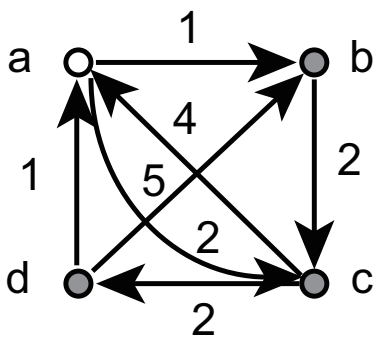
- L_i: the set of cycles in G that contain at most $|V|$ vertices and traverse the vertex i exactly once.

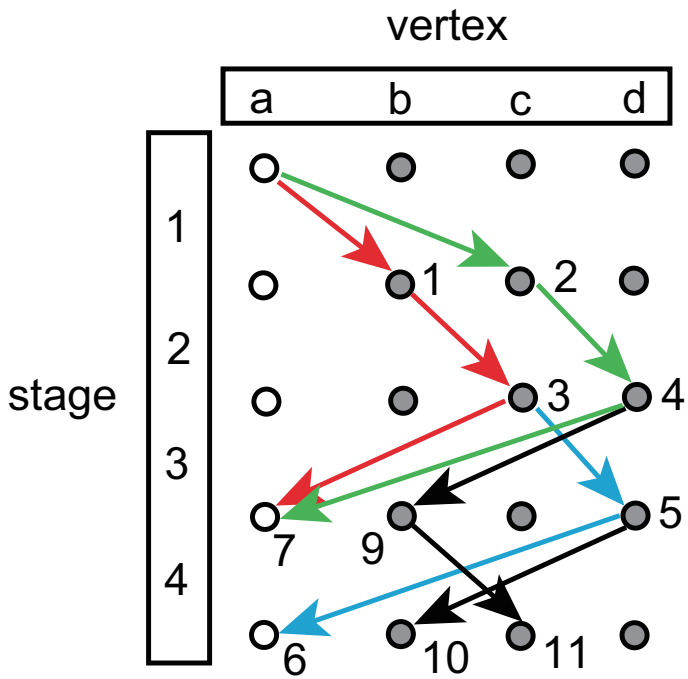
$$S \subseteq L \triangleq \bigcup_{i \in V} L_i.$$

- Defining $X_i \triangleq \max_{c \in L_i} w(c)$, we have,

$$X_L = \max_{i \in V} X_i.$$

Computer X_i by a Breadth First Search like Traversal





- $X_a = 7$.
- $X_L = max(X_a, X_b, X_c, X_d)$.
- Use statistical **sum** and **max** to compute X_i in statistical situation.

SGT-PC Algorithm

Subroutine VertexDist

Inputs

$G = (V, E)$: the graph. i: a vertex in V.

Outputs

X_i: maximum cycle weight for cycles passing i exactly once.

For all $j \in V$:

$$W_{ij}^1 \leftarrow \begin{cases} \max_{(i,j)\in E} w(i,j), & \exists (i,j) \in E, \\ -\infty, & \nexists (i,j) \in E. \end{cases}$$

For $k \leftarrow 2$ to $|V|$:

  For all $j \in V$:

$$W_{ij}^k \leftarrow \max_{\left(v \neq i\right) \wedge \left((v,j)\in E\right)} W_{iv}^{k-1} + w(v,j).$$

$X_i \leftarrow \max(W_{ii}^1, W_{ii}^2, \ldots, W_{ii}^{|V|})$.

Algorithm SGT-PC

Inputs

$G = (V, E)$: the graph.

Outputs

X_L: maximum cycle weight in L

For all $i \in V$:

  $X_i \leftarrow$ VertexDist(G, i).

$X_L \leftarrow \max_{i \in V} X_i$.

Algorithm Complexity

Theorem

Assume that each random variable requires $O(R)$ storage and each **sum** *or* **max** *operation requires $O(T)$ time.*

- *The time complexity of the SGT-PC algorithm is $O(T|V|^2|E|)$;*
- *The space complexity is $O(R|V|)$.*

Practical Implementation Considerations

- Graph decomposition by strongly connected components(SCC).
- Limiting number of traversal stages in VertexDist.

Graph Decomposition by SCC

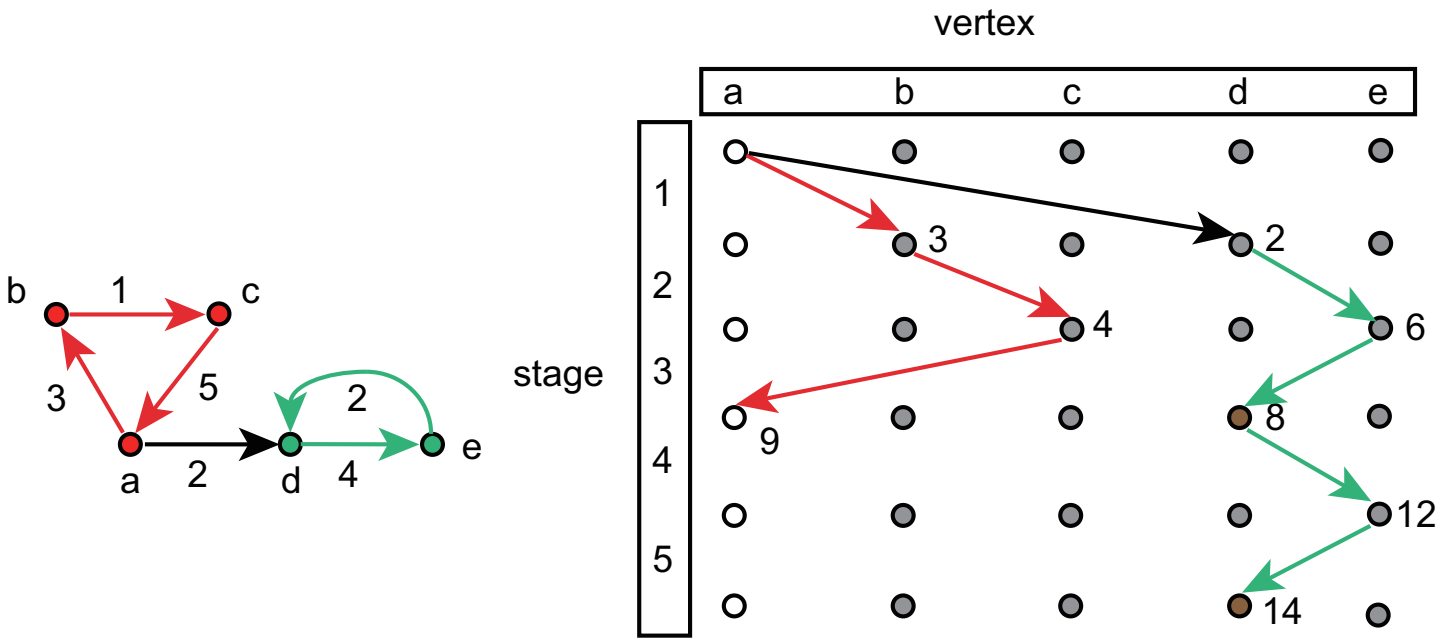
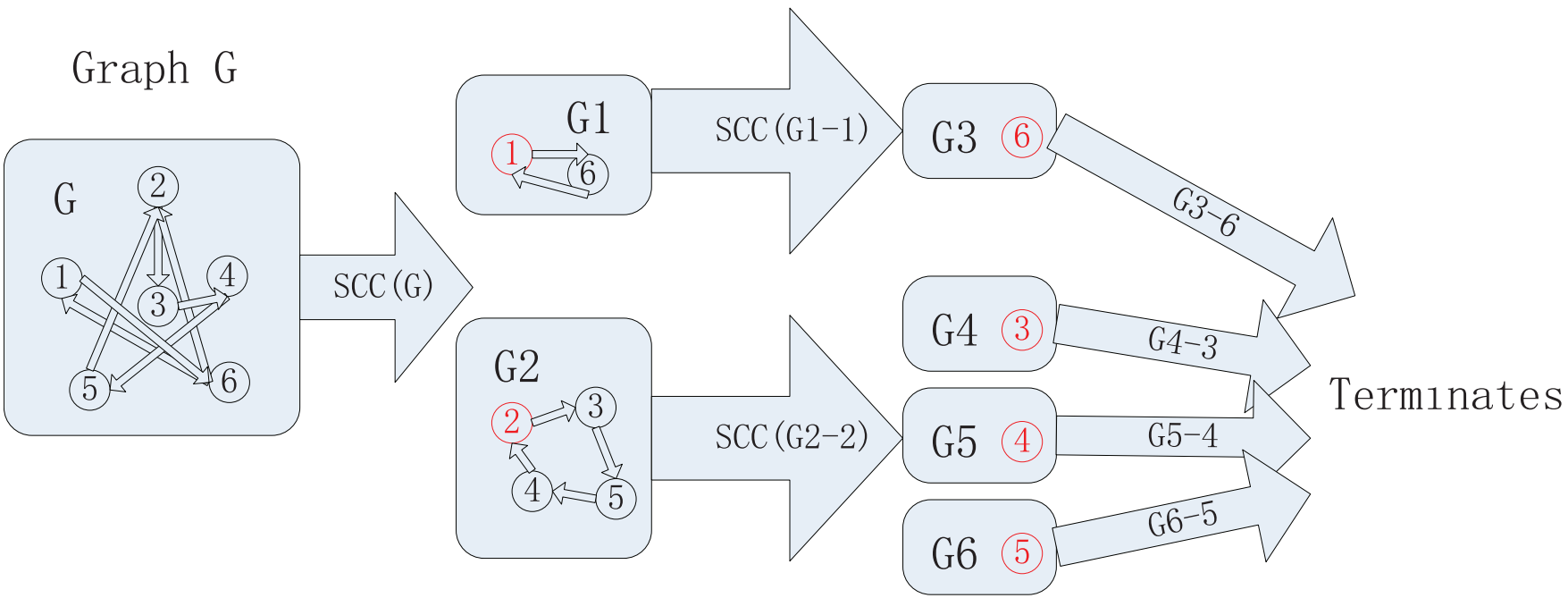


Figure: A graph with a path never traveling back to *a*.

The SGT-PC$_{SCC}$ Algorithm



Q: manage the data structure by storing the discovered SCCs but not processed.

Limiting Number of Traversal Stages

- Trade-off between solution accuracy and running time.
  - A bound N to the number of stages;
  - Some cycles could be missed.
- Motivation:
  - If the missed cycle is not a simple cycle, the solution accuracy will not be affected;
  - Time borrowing will tolerate acceptable variations.
- The designers to determine the bound N.

Limiting Number of Traversal Stages

- Trade-off between solution accuracy and running time.
  - A bound N to the number of stages;
  - Some cycles could be missed.
- Motivation:
  - If the missed cycle is not a simple cycle, the solution accuracy will not be affected;
  - Time borrowing will tolerate acceptable variations.
- The designers to determine the bound N.

Limiting Number of Traversal Stages

- Trade-off between solution accuracy and running time.
  - A bound N to the number of stages;
  - Some cycles could be missed.
- Motivation:
  - If the missed cycle is not a simple cycle, the solution accuracy will not be affected;
  - Time borrowing will tolerate acceptable variations.
- The designers to determine the bound N.

Outline

Experiments

- Experimental sequential circuits: ISCAS89 benchmark.
  - Each flip-flop is replaced by a pair of transparent latches.
  - One is moved among the combinational path while preserving the circuit functionality.
  - Two-phase clock is assigned to the latches.
- Generate gate delays:
  - A nominal delay for a gate is equal to the number of its fanouts.
  - Assign a standard deviation to each gate based on spatial variations (within $20 - 30\%$ of its nominal delay).
- Latest latch timing graph: generated by applying block-based SSTA techniques.
- Chen's PCycle, SGT-PC$_{SCC}$, and Monte Carlo simulation: implemented in C++, compiled by GCC version 3.4, and run on a Linux PC with a 2.4GHz processor and 4.0GB memory.

Experiments

Table: Comparison of Chen's PCycle, SGT-PC$_{SCC}$, and Monte Carlo simulation.

circuit			PCycle			SGT-PC$_{SCC}$			Monte Carlo	
name	$\|V\|$	$\|E\|$	yield%	time(s)	error%	yield%	time(s)	error%	yield%	time(s)
s27	7	21	97.37	0.02	0.11	97.37	0.01	0.11	97.26	0.34
s208.1	27	113	99.16	0.08	0.54	98.90	0.02	0.28	98.62	3.45
s382	49	249	99.76	0.25	2.53	97.37	0.03	0.14	97.23	11.68
s420.1	50	283	98.37	0.25	-0.21	98.82	0.03	0.24	98.58	13.43
s526	53	263	94.96	0.27	0.05	94.81	0.12	-0.10	94.91	19.93
s832	83	336	97.99	0.57	-0.37	98.06	0.15	-0.3	98.36	26.99
s1196	127	391	97.77	0.76	-1.63	99.39	0.43	-0.01	99.40	68.34
s1423	159	2315	96.38	2.39	1.55	94.66	0.64	-0.17	94.83	246.14
s5378*	514	2697	97.19	26.06	1.15	96.37	3.16	0.33	96.04	1228.06
s13207*	1365	5714	97.92	197.20	1.93	96.15	38.28	0.16	95.99	7515.15
s13207.1*	1337	5673	99.93	204.94	1.05	98.98	35.66	0.10	98.88	7613.59
s15850*	1275	17901	98.38	542.25	0.80	98.19	47.05	0.61	97.58	15136.97
s35932*	3145	10838	97.12	977.72	-2.08	99.50	466.29	0.30	99.20	35704.76
s38417*	3419	34971	96.91	3083.51	-0.69	97.62	494.23	0.02	97.60	85445.45
s38584*	4253	26208	99.70	2906.01	0.70	99.31	968.06	0.34	98.97	93268.92

Circuits with a $*$ mark, limit the number of traversal stages. On average, the error of the SGT-PC$_{SCC}$ is 0.21%, while the error of the PCycle is 1.03%.

Outline

Conclusion

- SGT-PC:
  - Cover all cycles through a structural graph traversal;
  - Within $O(|V|^2|E|)$ number of statistical **sum** and **max** operations.
- Practical Implementation:
  - Decomposition technique: strongly connected components;
  - Heuristic approach: limit the region of graph traversal to allow designers to trade-off accuracy with running time.
- Further work:
  - Reduce the statistical **max** operation errors;
  - Bound the circuit yield.

Thank you!

Questions?

Appendix

Table: Comparison on the count of **sum** and **max** operations for Chen's PCycle and SGT-PC$_{SCC}$.

circuit	PCycle		SGT-PC$_{SCC}$	
name	sum#	max#	sum#	max#
s27	2173	856	765	249
s208.1	26352	16146	7235	2076
s382	102916	67148	24396	7633
s420.1	106078	72754	21036	7538
s526	116123	79674	152221	75059
s832	314128	200318	207288	73183
s1196	459642	261392	659943	202957
s1423	6823K	5478K	741K	339K
s5378*	13900K	9193K	1849K	512K
s13207*	88255K	52408K	540K	103K
s13207.1*	95489K	57157K	1467K	277K
s15850*	267978K	211626K	23694K	10540K
s35932*	274623K	167203K	119737K	30157K
s38417*	1571896K	1278600K	6976K	2350K
s38584*	1377417K	970265K	84463K	18272K

Circuits with a $*$ mark, limit the number of traversal stages.