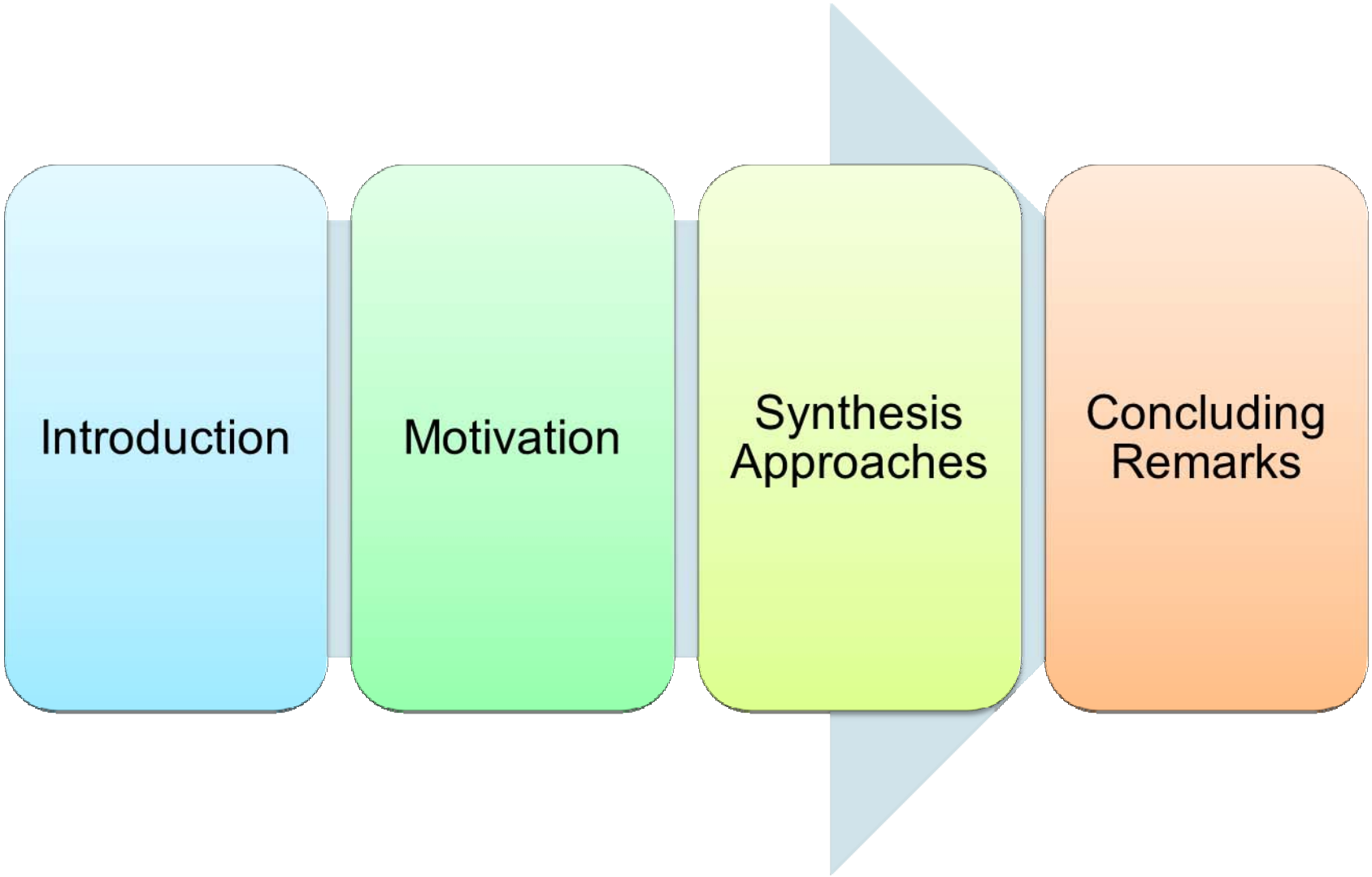
Symmetrical Buffered Clock-Tree Synthesis with Supply-Voltage Alignment

Xin-Wei Shih, Tzu-Hsuan Hsu, Hsu-Chieh Lee,
Yao-Wen Chang, Kai-Yuan Chao

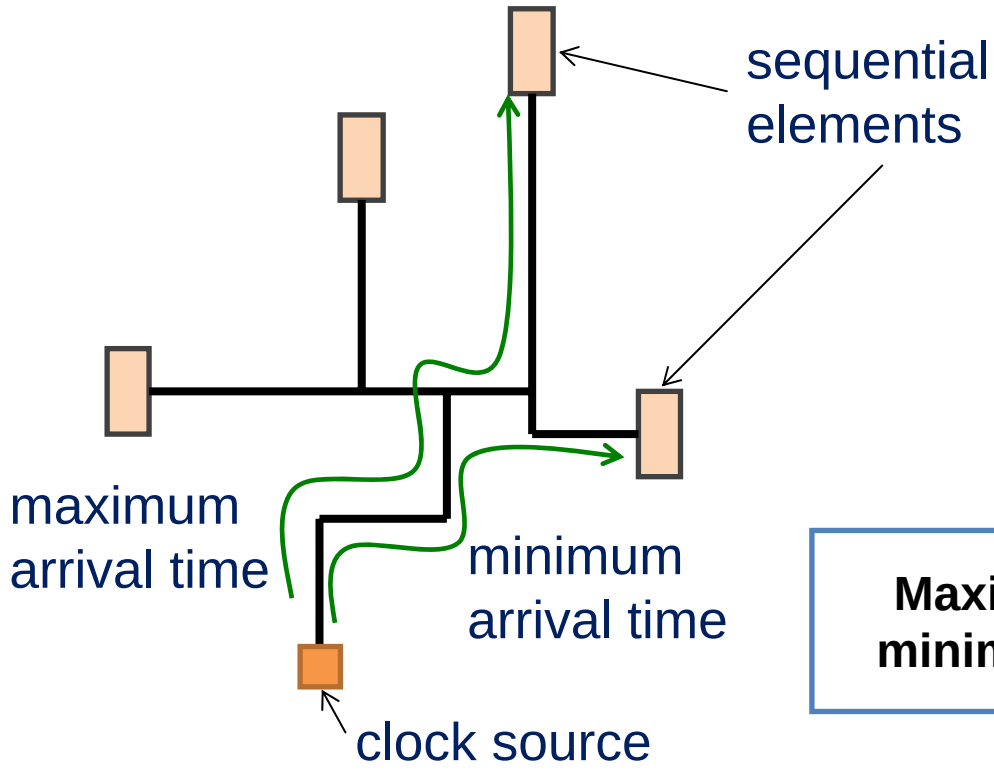
2013.01.24

Outline


Introduction
Motivation
Synthesis Approaches
Concluding Remarks

Clock Network Synthesis

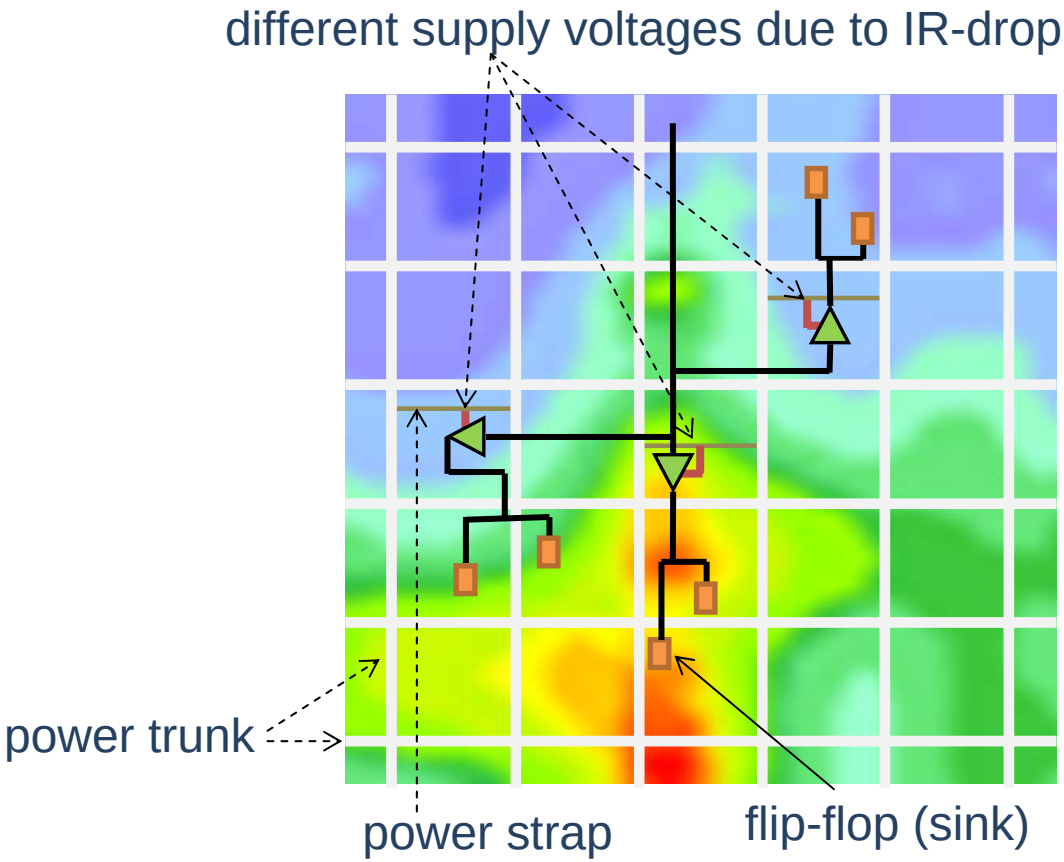
- Clock network connects sequential elements (sinks)
- Skew-minimized buffered clock-tree synthesis plays an important role for synchronous circuits



Maximum arrival time minus minimum arrival time → Skew

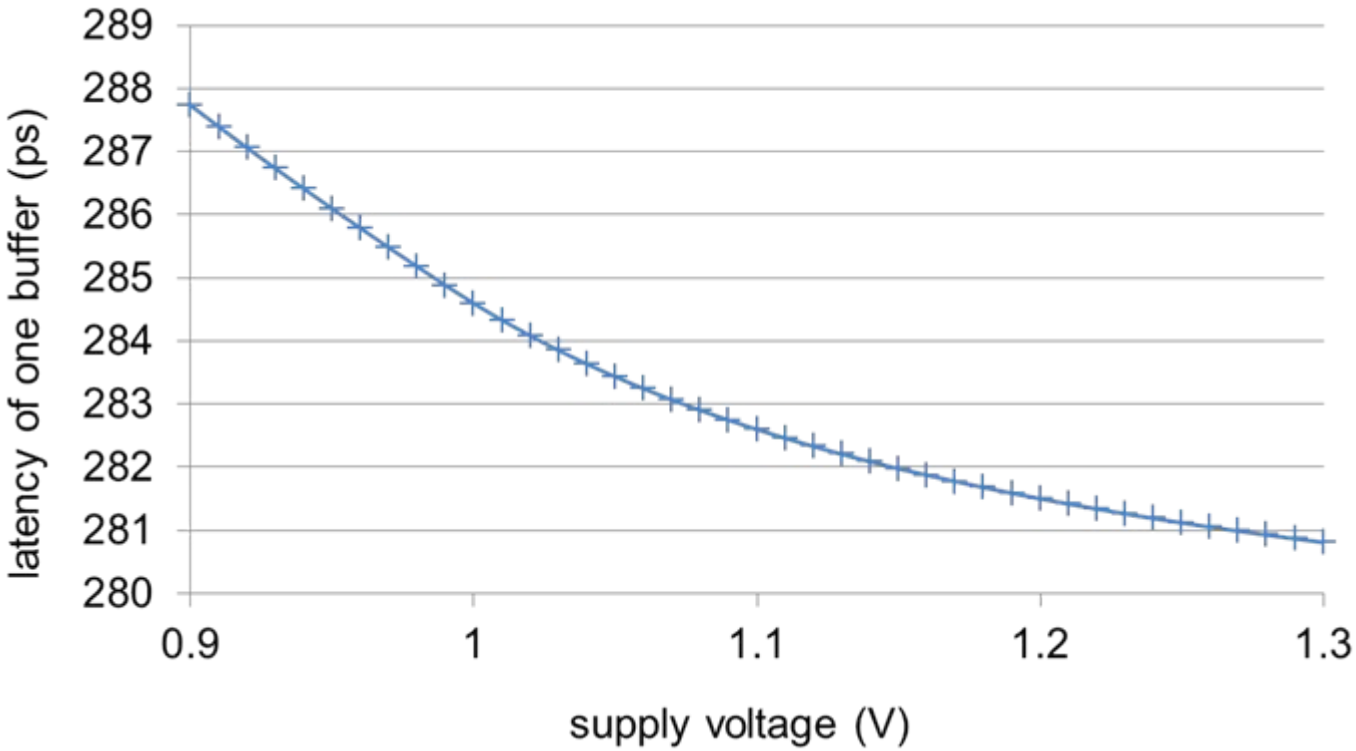
IR-Drop Effects on Clock Skew

- Buffer supply voltages could be different due to IR-drop
- If supply-voltage differences are not considered, the actual clock skew could be much worse than expected



IR-Drop Effects on Buffer Latency

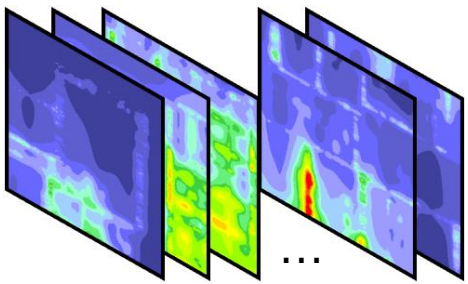
- IR-drop effects on power network change clock-buffer supply voltages → non-uniform supply voltages
- Experiments show that at least 0.2ps latency difference as the supply voltage changes with 0.01V for one buffer



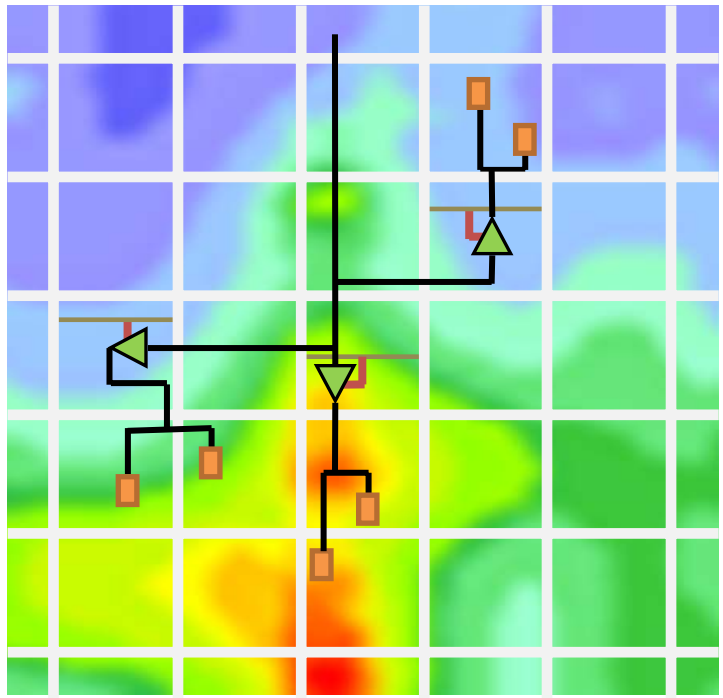
Problem Formulation

- Instance: given a set of clock sinks, a power network, a set of power-analysis data (the most timing-critical IR-drop map), a slew-rate constraint, and a library of buffers
- Question: construct a buffered clock tree and minimize its skew, subject to no slew-rate violation

IR-drop maps under different workloads

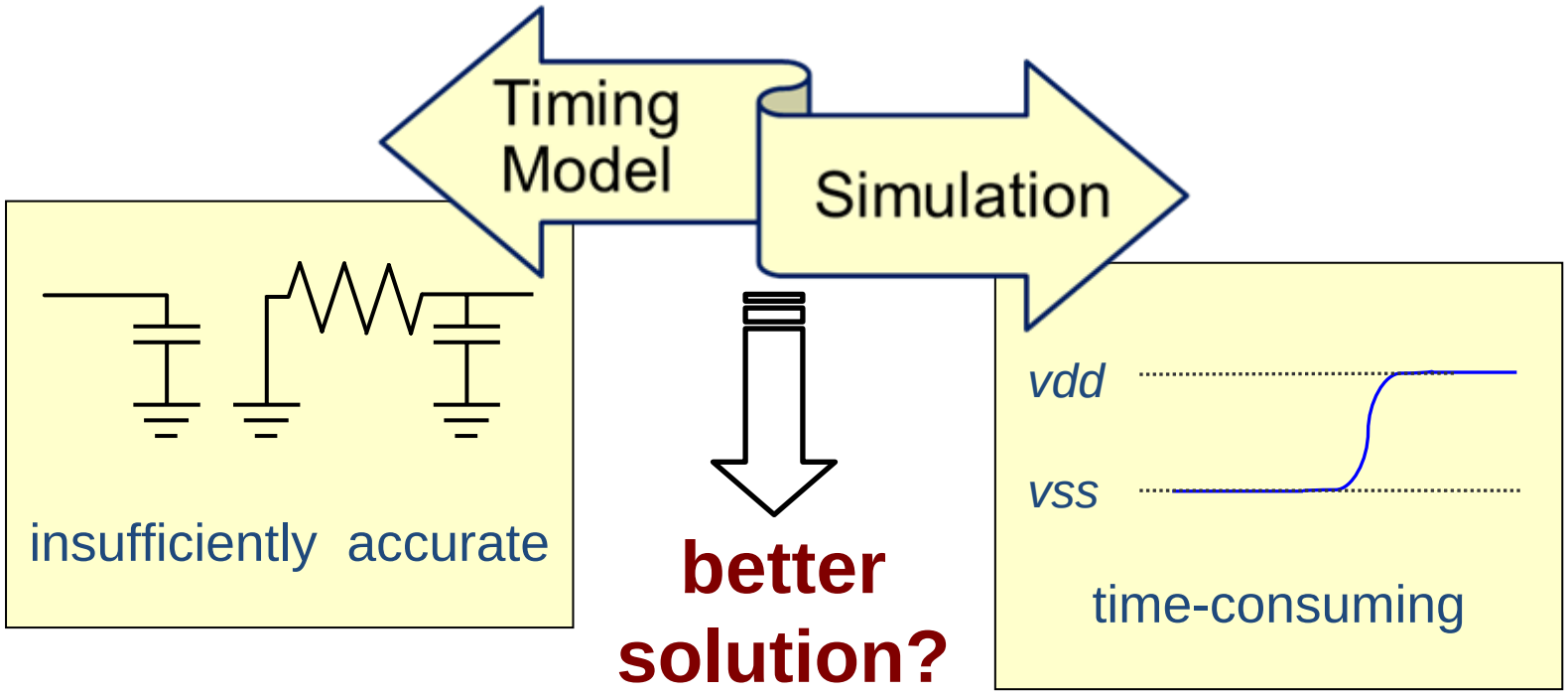


most timing-critical map



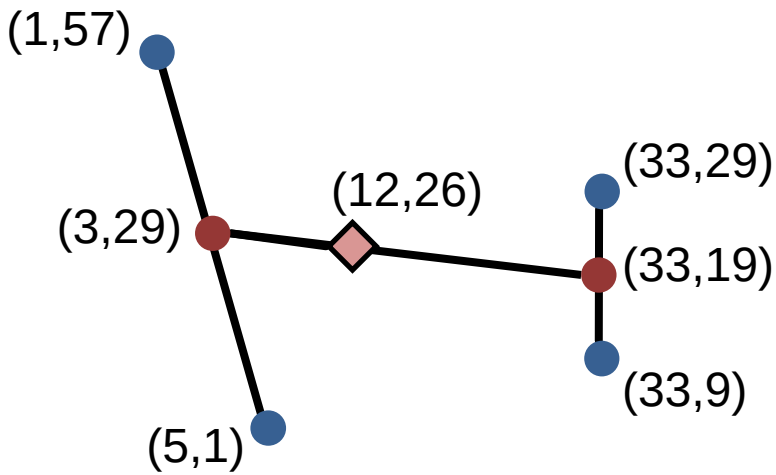
Conventional Clock Tree Synthesis

- Timing models (e.g., Elmore delay) are insufficiently accurate to assist in minimizing skew
- Synthesis running time becomes prohibitively long if simulations are embedded

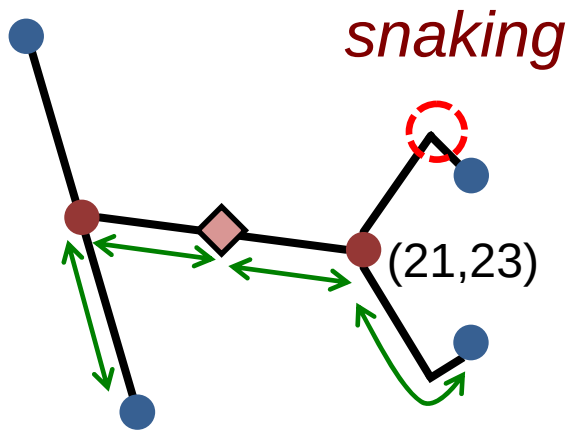


Structural Optimization: Symmetrical Structure

- Path configurations from root to sinks are similar
- Identical number of branches, identical wirelength, and identical inserted buffers hold at each tree level
  - Realization needs neither timing models nor simulations



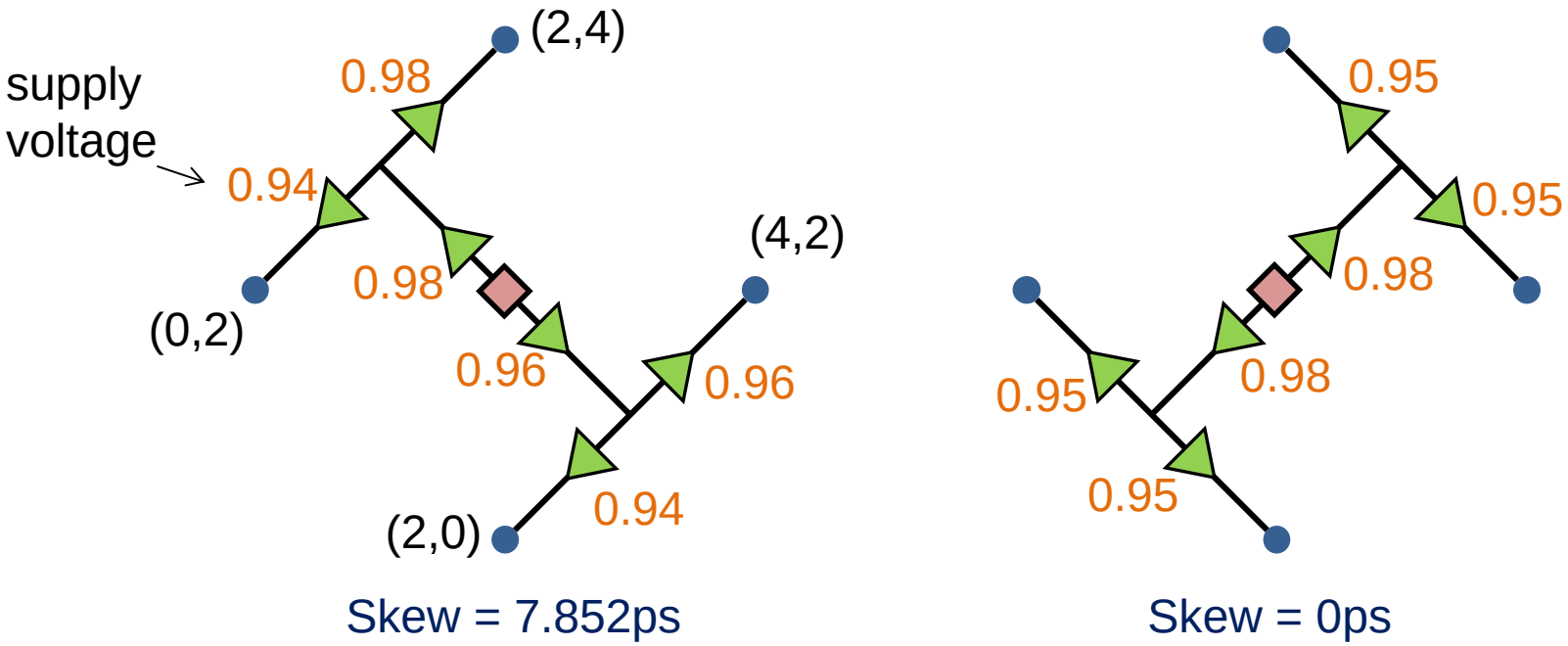
0ps skew (Elmore delay)
0.123ps skew (simulation)



0ps skew (Elmore delay)
0ps skew (simulation)

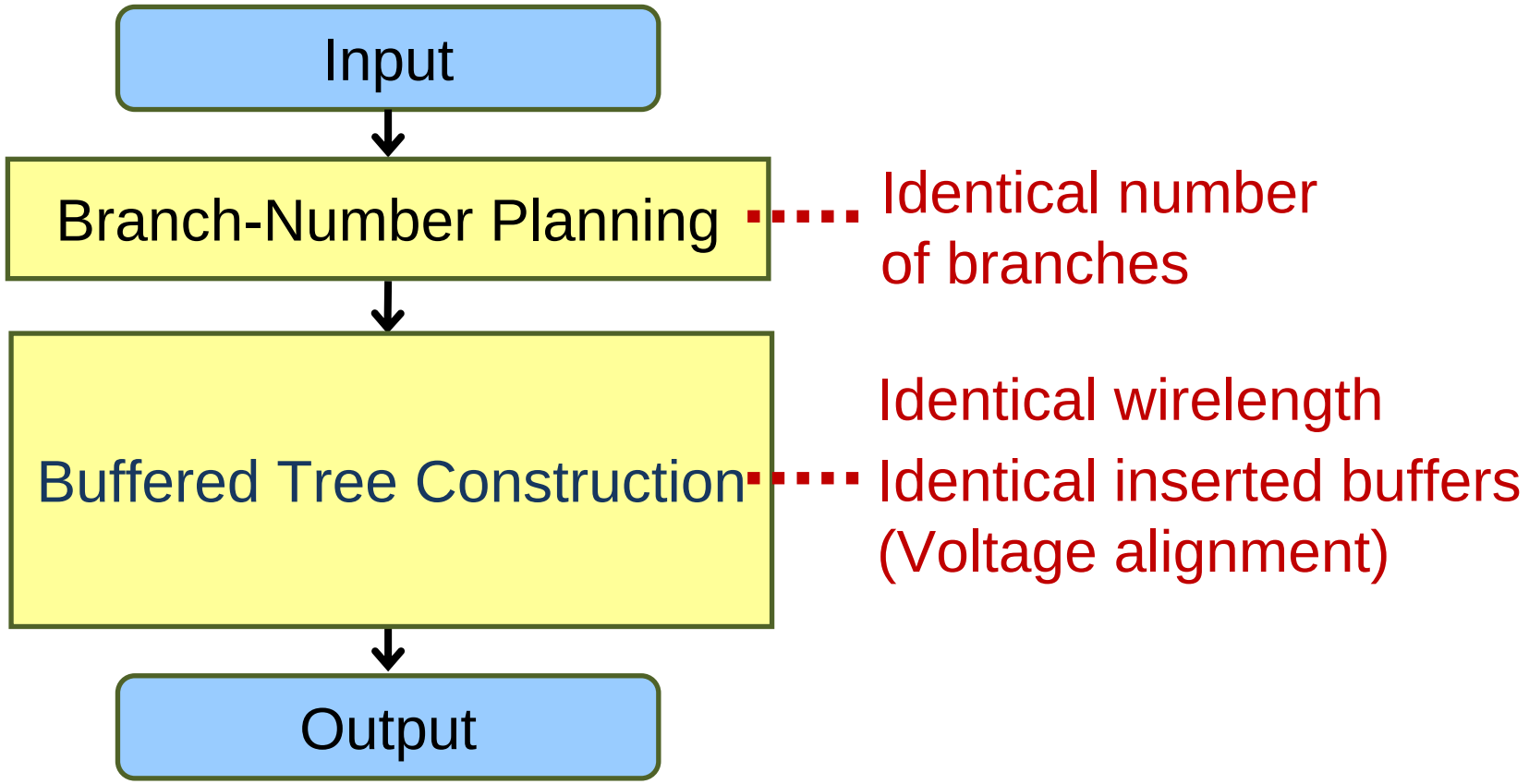
Motivation of Voltage Alignment

- Although clock trees are symmetrical, non-uniform supply voltages might affect skew
- Proposed solution: *voltage alignment* (minimizing voltage difference for the same tree level)



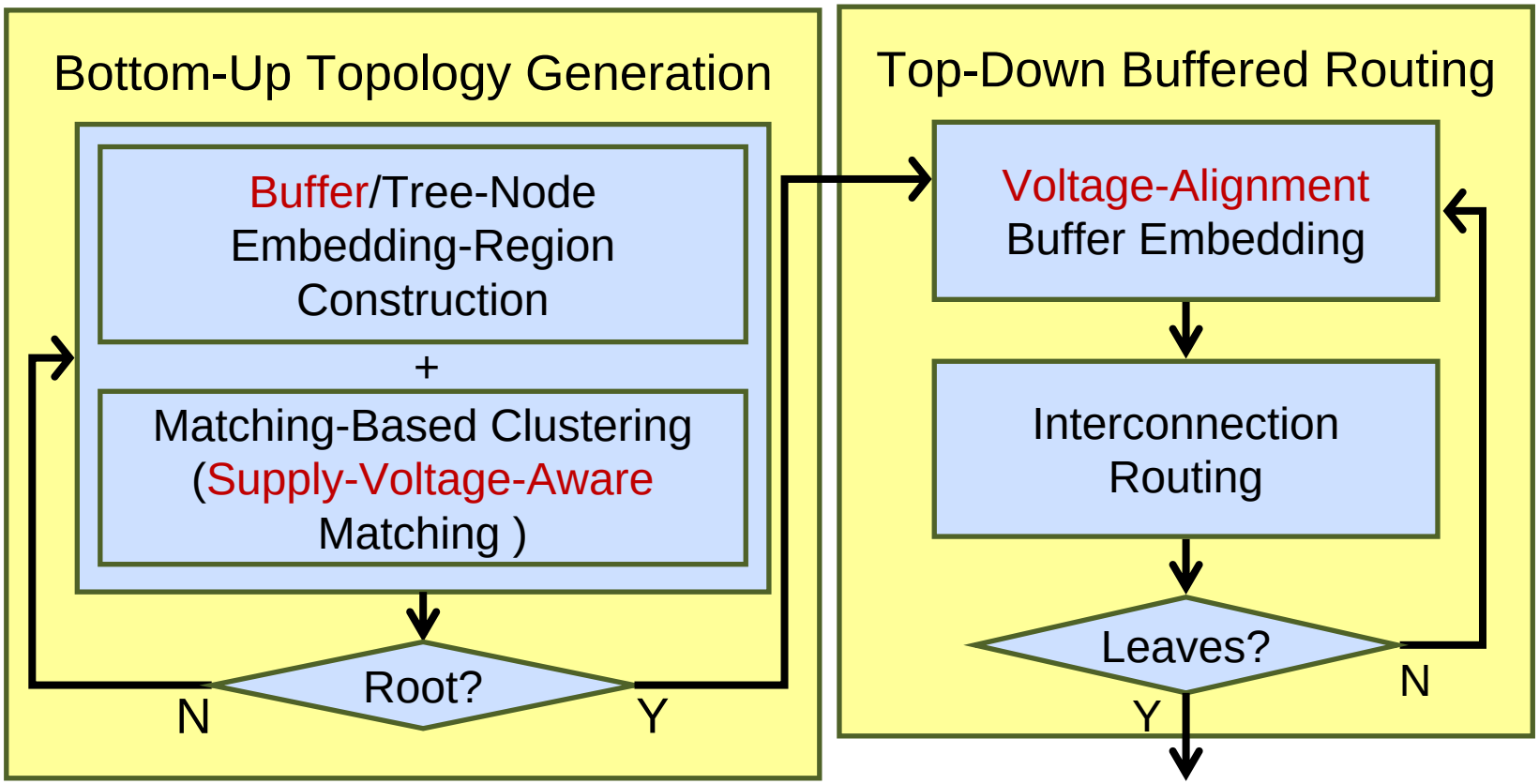
Supply-Voltage-Aware Clock Tree Synthesis

- Plan branch numbers by factorization [Shih et al., DAC'10]
- Propose simultaneous tree construction and buffer insertion to realize voltage alignment



Buffered Tree Construction

- First build a topology that benefits voltage alignment
- Then place buffers where the used voltages are aligned



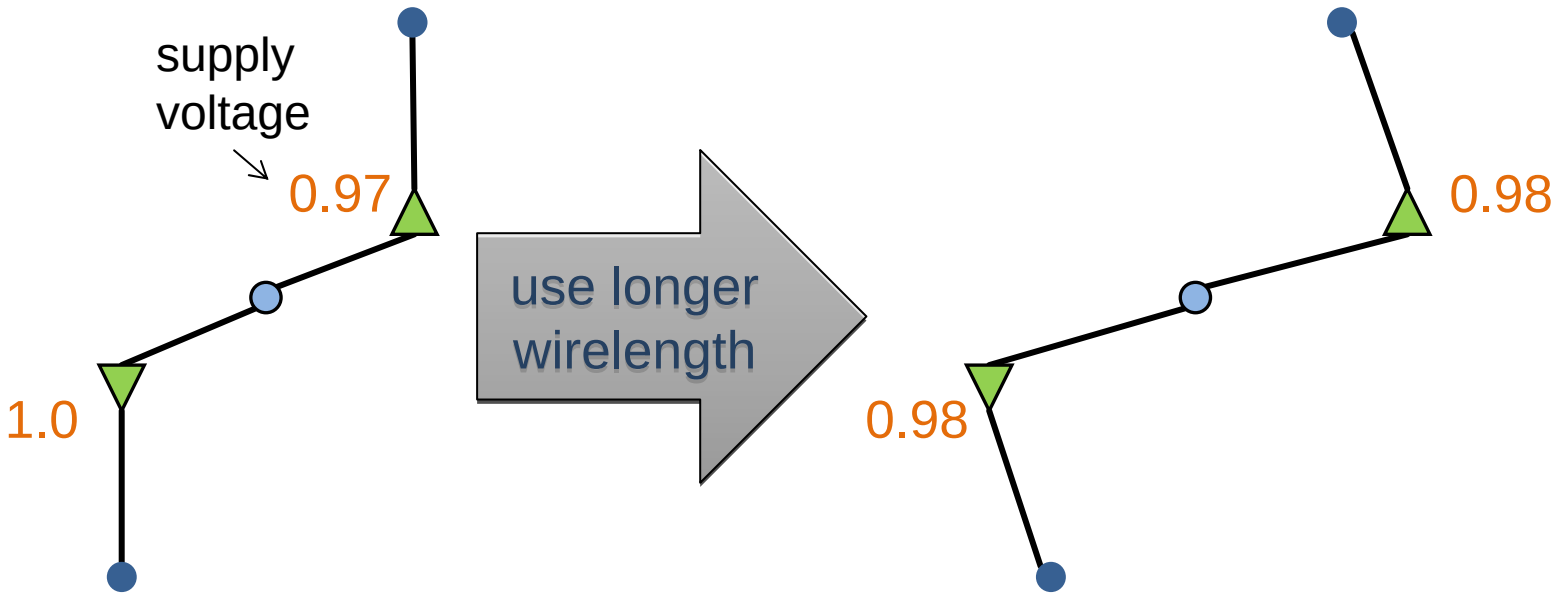
Matching-Based Clustering Algorithm

- **Minimum-Bottleneck Maximum Matching (MBMM)**
  [Shih et al., ICCAD'10]
  - Perform optimizations on a graph
    - Vertex represents a sink or a subtree
    - Edge defines a cluster of subtrees
    - Edge cost is the wirelength of the defined cluster
  - Select edges to form *perfect matching* (i.e., each vertex must be matched)
    - Selected edges indicates the clustering
  - Update the graph iteratively by selecting and removing edges
  - Minimize the bottleneck cost (maximum cost) of selected edges

Achieve the optimum bottleneck cost for a problem instance with a branch number of two (commonly used branch number)

Trade-Off: Voltage Alignment and Wirelength

- Longer wirelength implies that more positions and thus more voltages can be selected for buffers
- Voltage difference for some tree level might be reduced by snaking wires



What If Directly Use MBMM for Clustering?

- New edge costs (for trade-off) can be modeled based on a combined cost of wirelength and voltage-alignment

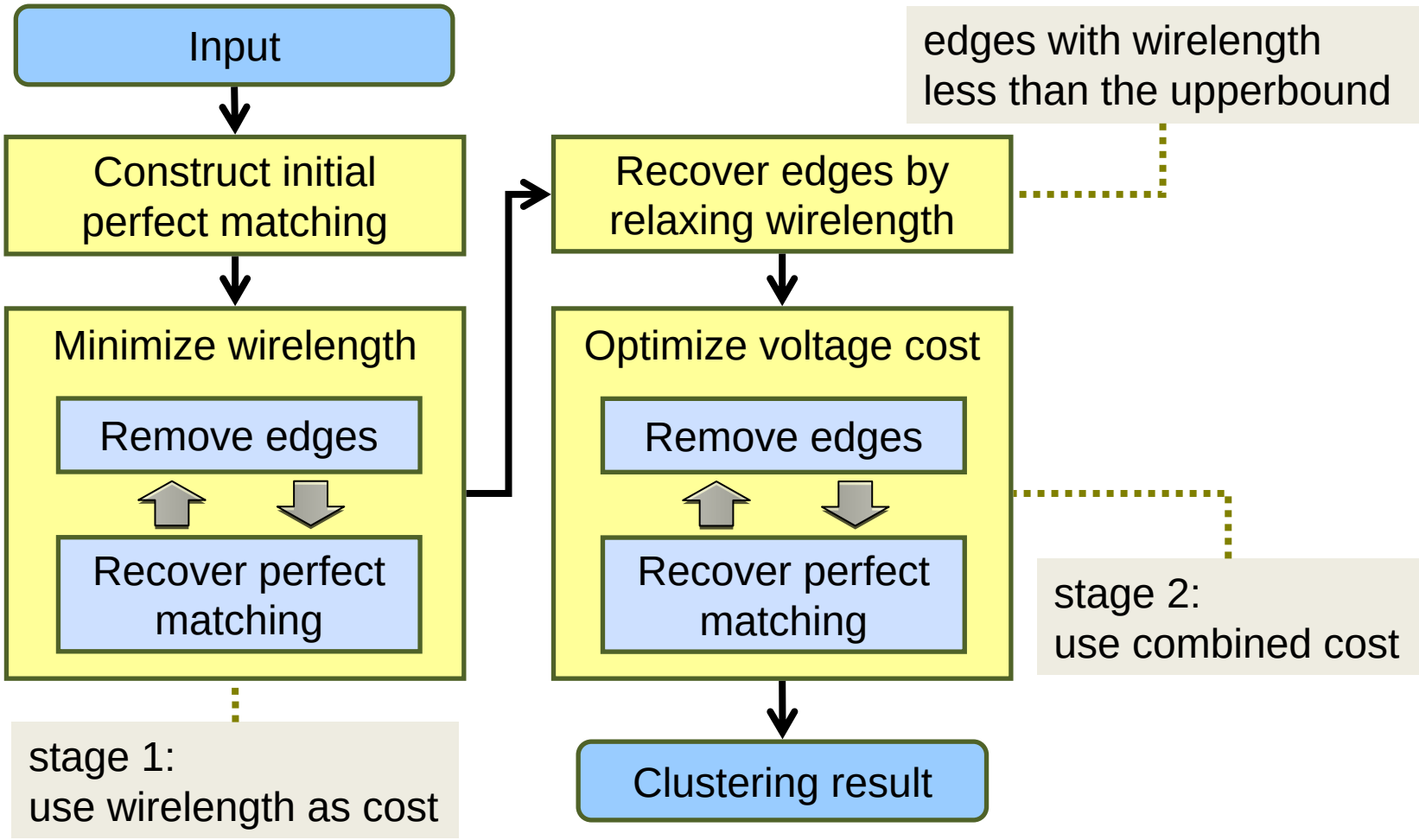
$$Cost(e) = 0.5 \cdot Wirelength(e) + 0.5 \cdot VoltageAlignment(e)$$

- Only applying the model to MBMM has drawbacks in prior iterations
  - Extremely long wirelength is not acceptable so that involving voltage cost is not necessary but time-consuming
  - Wirelength is not stable so that voltage cost is imprecise

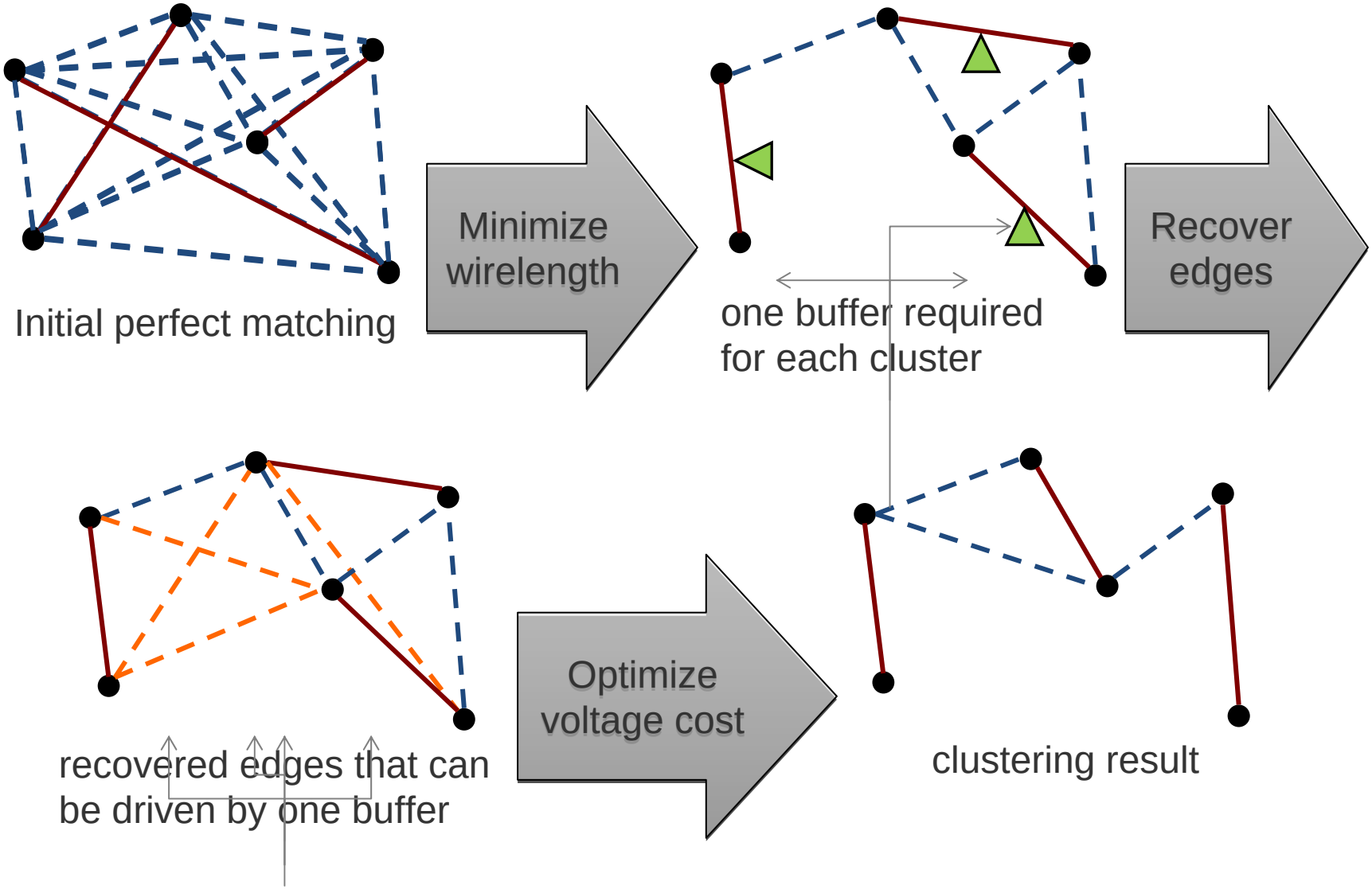
Two-stage approach!
Minimize wirelength first and then optimize the combined cost for a certain range of wirelength

Proposed Supply-Voltage-Aware Matching

- Wirelength upper bound of stage 2 is determined based on the required buffer number of stage 1



Supply-Voltage-Aware Matching Example



Voltage Alignment Cost Model

- Locate possible positions for buffers to identify corresponding candidate voltages: voltage interval
- Compute an expected voltage from all voltage intervals
- Define the cost as the mismatch between voltage intervals and the expected voltage

$$m(v^e, v_1, v_2) = \begin{cases} |v^e - v_1| & \text{if} \quad v^e < v_1, \\ 0 & \text{if} \quad v_1 \le v^e \le v_2, \\ |v^e - v_2| & \text{if} \quad v^e > v_2. \end{cases}$$

expected voltage

minimum value of interval

maximum value of interval

Expected Voltage

- The expected value results in the minimum sum of mismatch
- Formulation:

intervals

$$\min_{v^e \in \Re} f(v^e) = \sum_{i=1}^{n} m(v^e, v_{i_1}, v_{i_2})$$

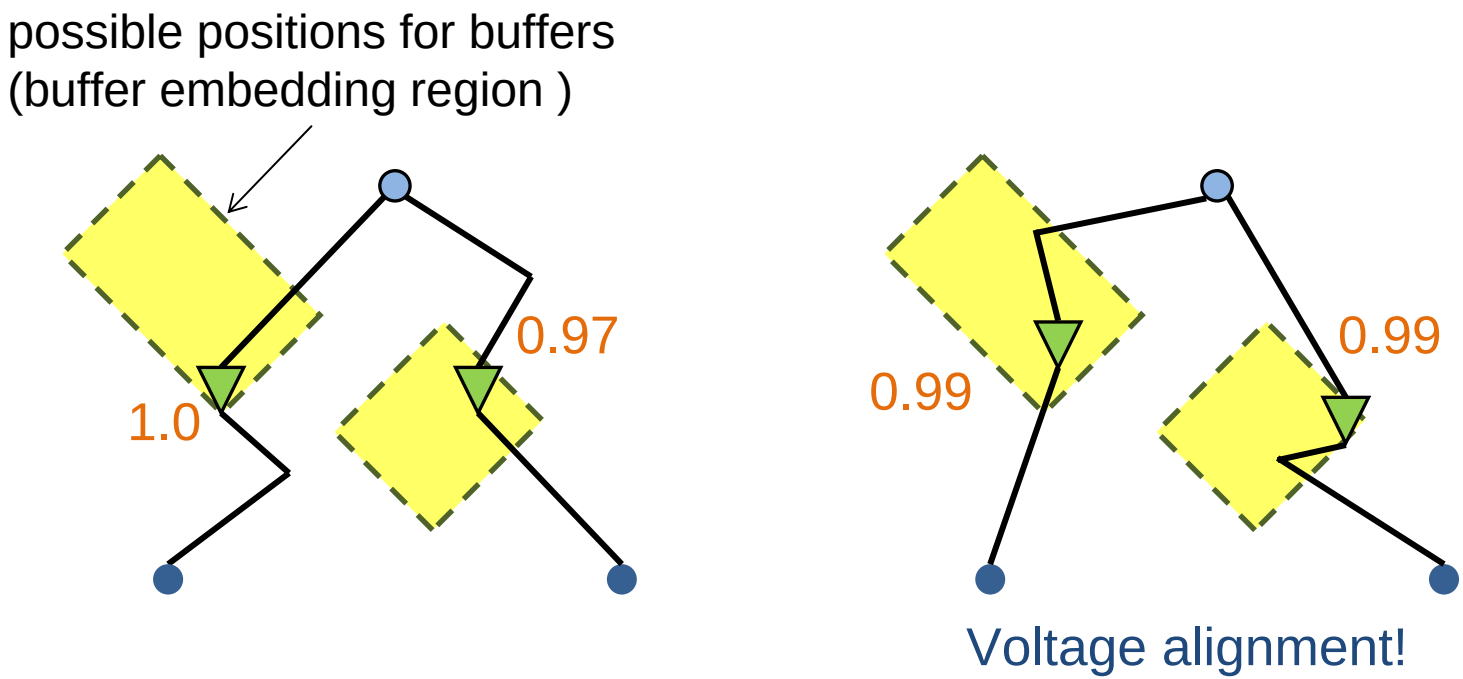
minimum and maximum values of i-th interval

- Determining expected voltage for n voltage intervals can be solved by transforming the problem into finding the median in $2n$ end-points of the n voltage intervals

$$\min_{v^e \in \Re} \left(\sum_{j=1}^{2n} \left| v^e - u_j \right| \right)$$, where $$u_i = v_{i_1}, u_{n+i} = v_{i_2}, \forall i \le n$$

Voltage-Alignment Buffer Embedding (Placing)

- Compute the expected voltage for buffers of the same tree level
- Place each buffer of the same tree level at a position with the voltage closest to the expected voltage



Experimental Setting

- Platform: 2.93 GHz Intel Xeon workstation
- Some ISPD'09 and ISPD'10 contest benchmarks
- Power networks are generated for benchmarks
  - 10 x 10 grids (11 horizontal and 11 vertical power trunks)
  - Random current value for each power node to simulate the real-chip situation
  - 5% IR-drop tolerance of the nominal supply voltage (a common constraint for high-end designs [Subramaniam, Power management for optimal power design, EDN'10])
  - Nominal supply voltage: 1.0V (original setting of ISPD'10 contest benchmark)
  - Available supply voltages: between 0.95V and 1.0V

Comparison among Different Flows

case	#sinks	Unware			Hybrid			Ours		
		skew (ps)	usage (fF)	CPU (s)	skew (ps)	usage (fF)	CPU (s)	skew (ps)	usage (fF)	CPU (s)
ispd'09-f11	121	3.02	69743	0.02	2.09	69743	0.02	1.69	72413	0.02
ispd'09-f12	117	3.64	61462	0.02	3.53	61462	0.02	1.47	64413	0.02
ispd'09-f21	117	9.10	78329	0.02	4.69	78329	0.02	3.15	78790	0.02
ispd'09-f22	91	10.26	43109	0.01	4.19	43109	0.01	2.40	43905	0.01
ispd'10-07	1915	21.26	152231	10.12	7.65	152231	10.53	5.86	154255	15.15
ispd'10-08	1134	43.03	124132	2.02	8.45	124132	2.02	4.41	125165	2.45
avg cmp		**4.13**	**0.98**	**0.92**	**1.68**	**0.98**	**0.92**	**1.00**	**1.00**	**1.00**

Skew is efficiently improved with marginal capacitance overhead as the complete flow considers the voltage alignment

Conclusion

- We present the first work of supply-voltage-aware buffered clock tree synthesis
- We minimize the clock skew by aligning used supply voltages for each level during clock-tree synthesis
- Experimental results shows that our proposed approach is both efficient and effective
- In particular, our work provides a key insight into the importance of handling practical design issues (such as IR-drop) for real-world clock-tree synthesis

Thank You!