Discrete Buffer and Wire Sizing for Link-Based Non-Tree Clock Network

Rupak Samanta, Jiang Hu and Peng Li

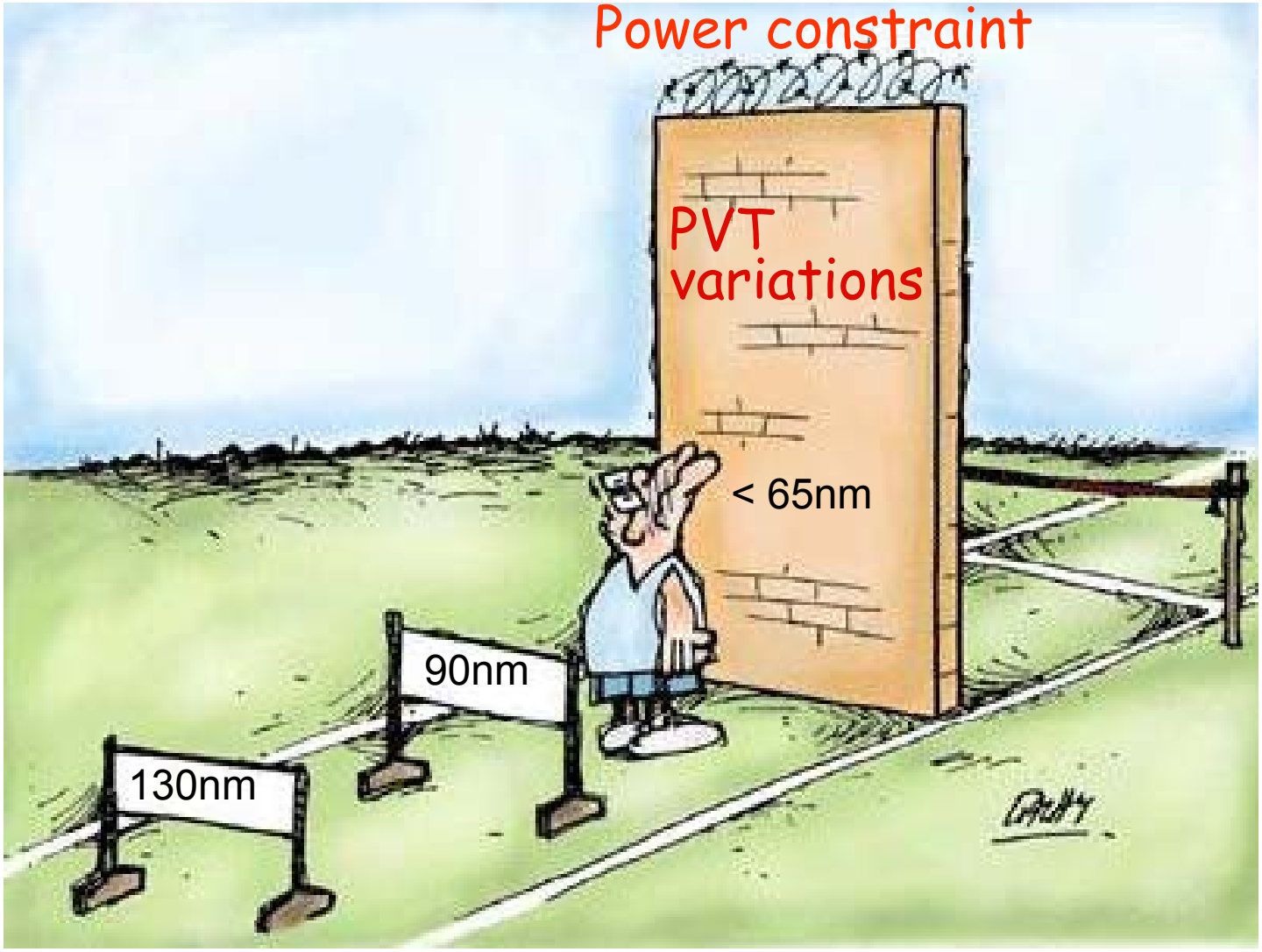
Department of Electrical and Computer Engineering

Texas A&M University

Outline

- Introduction
- Preliminary of Support Vector Machine
- Sizing Algorithm
- Experimental Results
- Conclusion

Challenges



Challenges in Clock Network

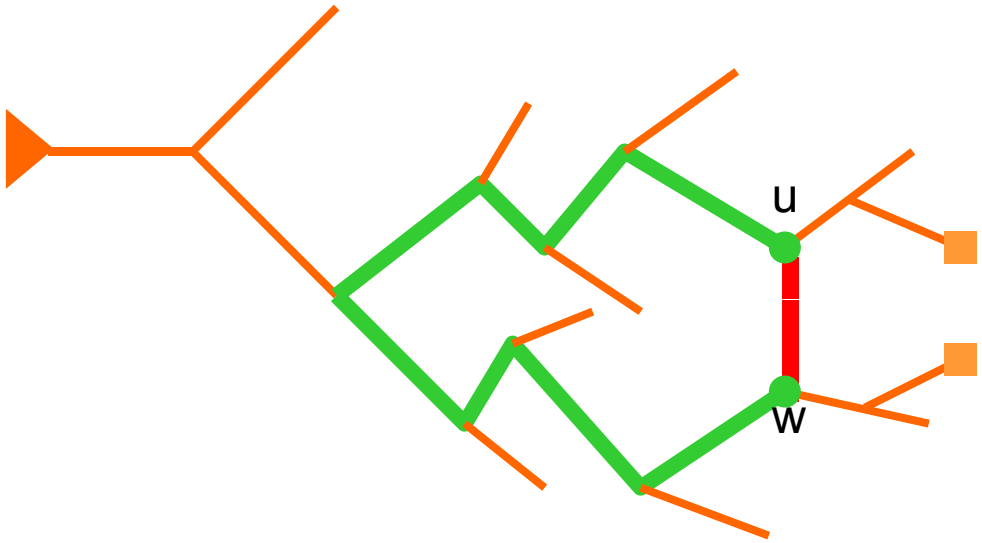
- Clock network is a sub-circuit that involves both the challenges – variability and power consumption
- A well known approach for skew tolerance to variation is clock mesh – large wire/power overhead
- Link-based non-tree clock network provides trade-off between variation tolerance and power overhead

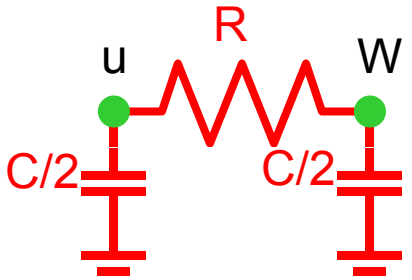
Our Objective

“To investigate optimizing link-based clock network through discrete buffer and wire sizing”

Review of Link Insertion

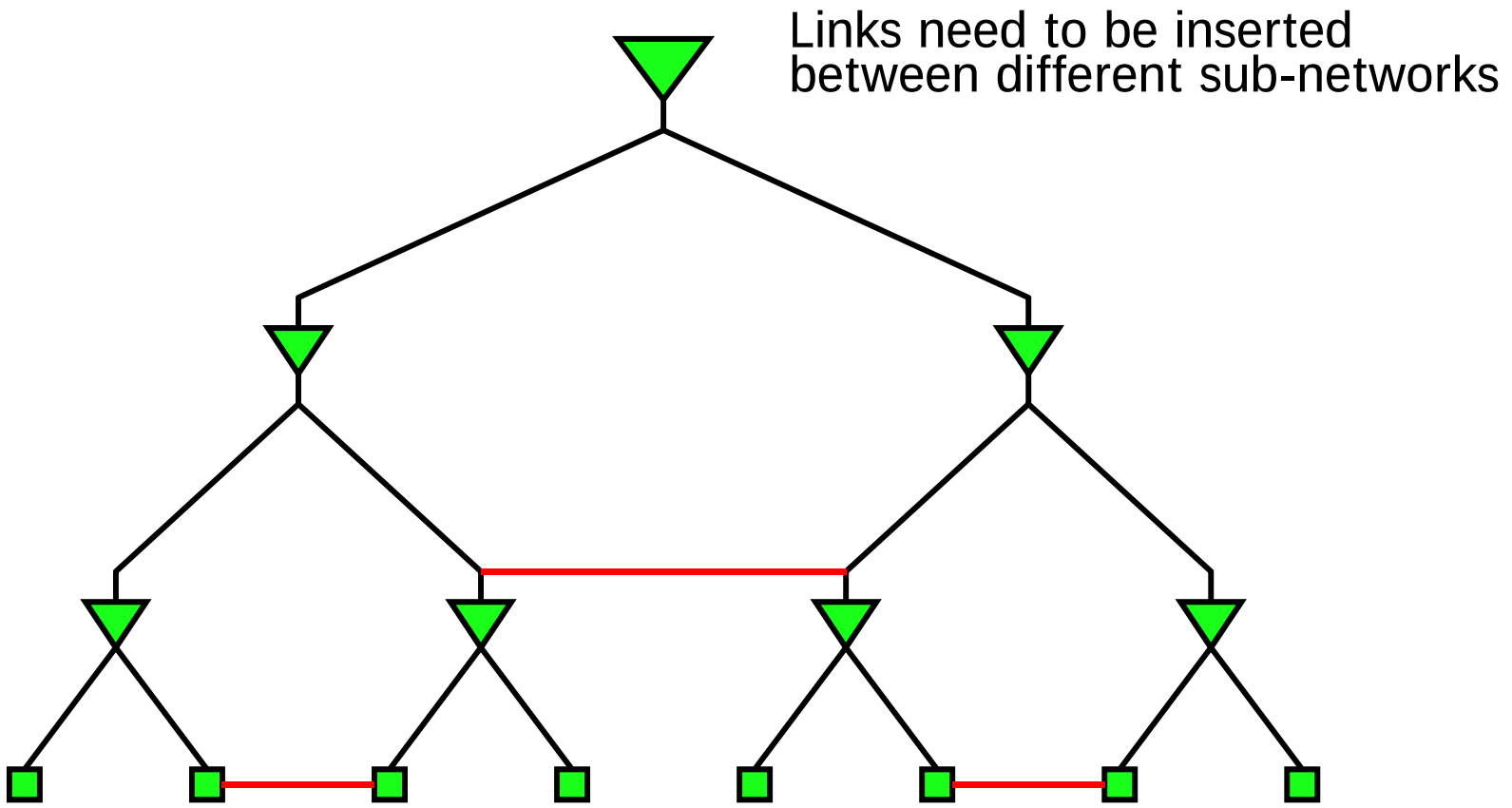
- A. Rajaram, J. Hu and R. Mahapatra, *DAC04*
- Add links such that skew due to variations is reduced AND nominal skew is unaffected
- Link = link_capacitors + link_resistor





Link Insertion for Buffered Clock Tree

- G. Venkataraman *et.al.* ICCAD05



Motivation

- There is almost no work on optimizing clock network with cross links
- Most of the previous work on buffer and wire sizing
  - Elmore delay model
  - continuous sizing
- Elmore delay model is inaccurate and differs by large amount when compared with SPICE
- The number of buffer and wire options are small, rounding continuous sizing result in significant errors

Our Contributions

- Support Vector Machine (SVM) is explored
  - handle the complex delay model issue
  - provide guidance for discrete optimization in large design space
- Proposed a two stage hybrid optimization approach
  - Discrete sizing
  - Using accurate delay model

Support Vector Machine(SVM)

- SVM is well suited for highly nonlinear and high-dimensional data
- SVM can operate in different modes
  - Classification
  - Regression
  - Ranking
- For a set of M training data set

  (x_1, y_1) ……(x_m, y_m), the regression model

$$f(x) = \Sigma \alpha_i K(s_i, x)$$

- The kernel function $K(s_i, x)$ corresponds to a dot product in certain feature space

Skew Quality Function

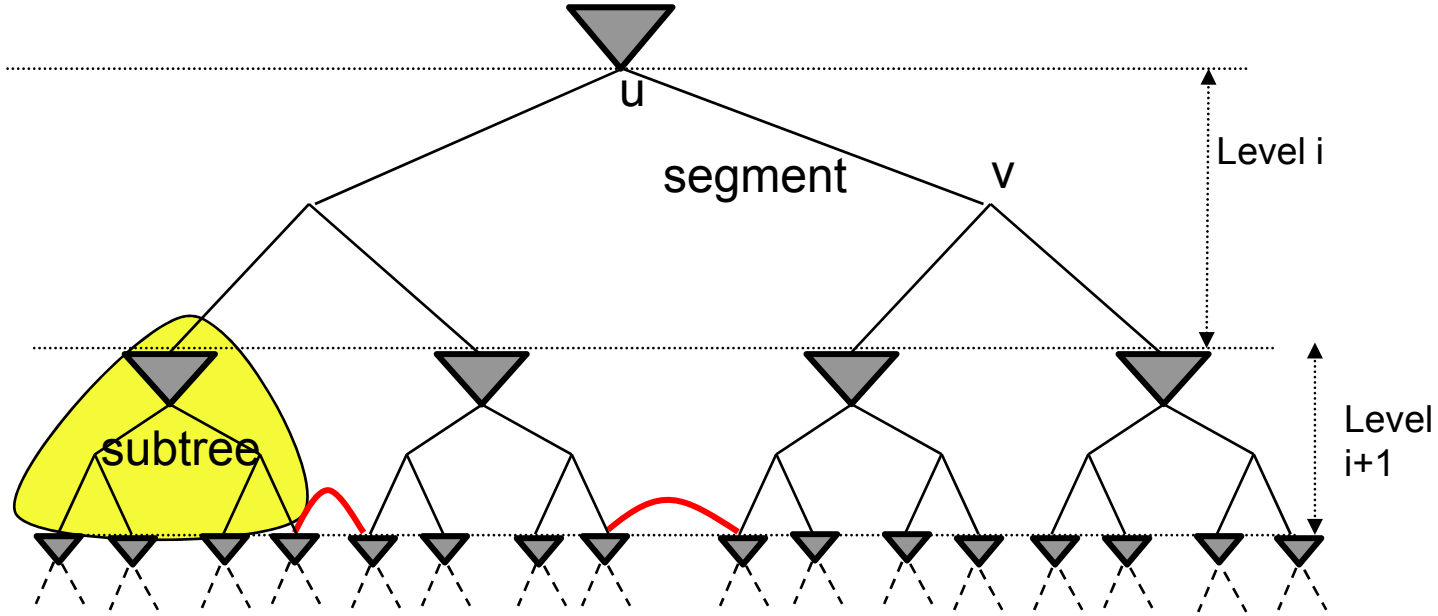
$$Q = \sum (t_i - t_j)^2$$

- t_i is clock delay at leaf node i
- An overall function that penalize large skew
- Clock delay is obtained through SPICE simulations
- The data is applied to train SVM model of Q
- Once an SVM model is built, it is utilized repeatedly, then the training cost is amortized

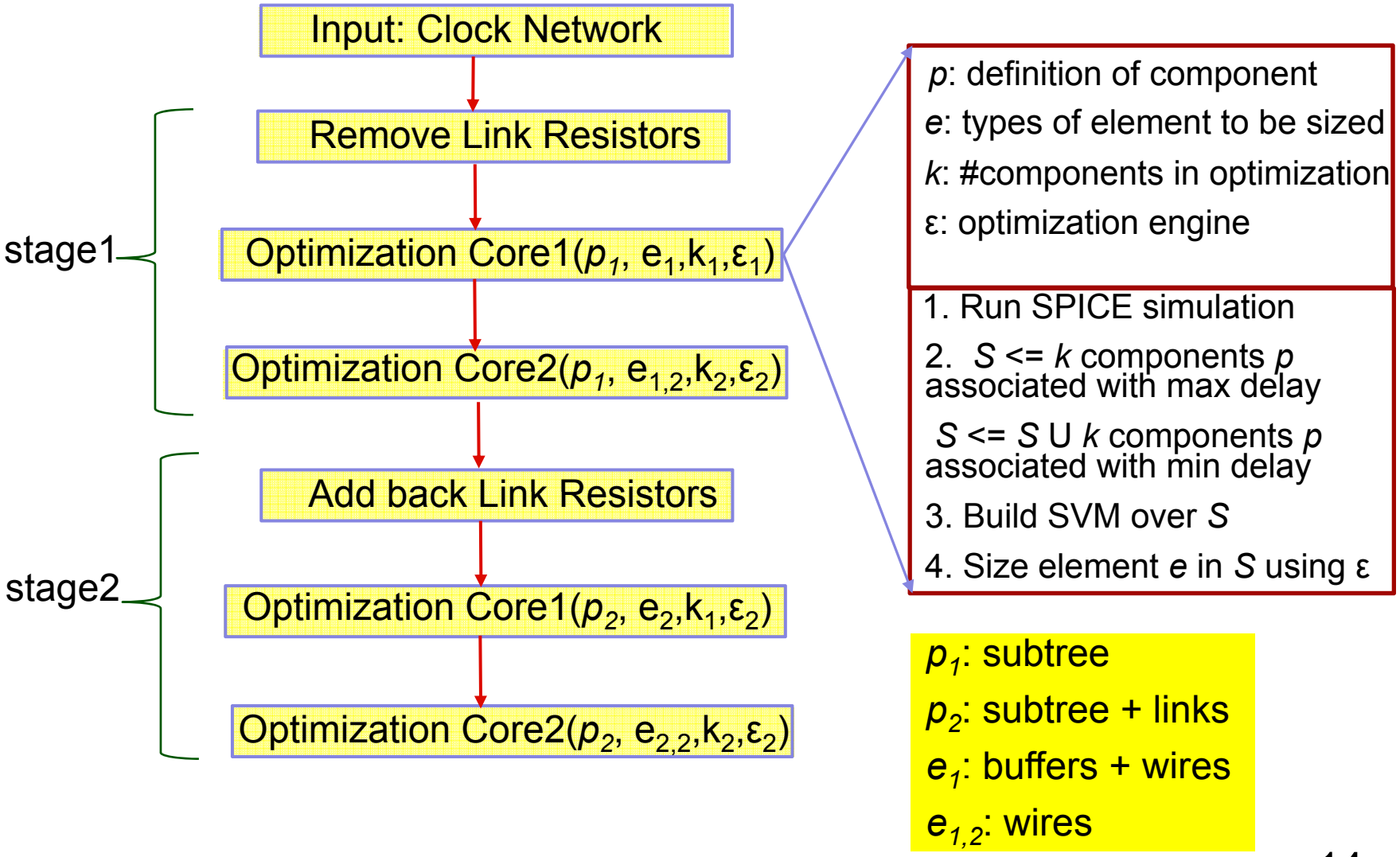
Sizing Algorithm

- The goal of the buffer and the wire sizing is to minimize the global skew
- Our approach is to iteratively optimize a portion of the given clock network
  - Compared to simultaneously optimizing the entire network our approach is more practical
  - Compared to iteratively optimizing a single element, our approach is more efficient on finding a global solution

Buffered Clock Tree with Cross links



Optimization Flow



Optmization Stage I

Link Resistors are removed, Link capacitance retained

STEP 1

- Step 1 is done on a coarse level
- Each wire segment in a sub-tree is sized uniformly
- Only one variable is needed for the wire
- This is done to reduce the total number of variables in each sub-tree
- More sub-trees can be chosen for optimization

STEP 2

- Step 2 is on fine-grained and local level
- Individual wires in each sub-tree are sized differently
- For each sub-tree the number of variables is usually large
- The number of sub-trees chosen for optimization are smaller than Step 1

Optimization Engine for Stage I

Input: Set of Components to be sized

1. While (*improve*) {
2. Partition S into a set G of m groups
3. Obtain average leaf node delay t_{avg} of each group
4. Sort groups in G in non-decreasing order of t_{avg}
5. For *i* = 1 to m/2 {
6. while (*improve*)
7. Increase t_{avg} of g_i in G by δ
8. while (*improve*)
9. Decrease t_{avg} of g_{m-i+1} in G by δ
10. }
11. }

ILP

τ is error tolerance

$$\delta - \tau \le \Delta t_l \le \delta + \tau, \ \forall \text{ leaf node } i$$

$$\sum_{\forall b} x_{i,b} = 1, \ \forall \text{ buffer } i$$

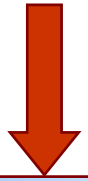
$$\sum_{\forall w} y_{j,w} = 1, \ \forall \text{ wire } i$$

$$x_{i,b} \in \{0,1\} \ \forall i,b$$

$$y_{j,w} \in \{0,1\} \ \forall j,w$$

Optimization Stage II

Link Resistors added back



- The clock network topology becomes non-tree
- Similar to stage I, this stage consists of two steps of optimizations
- Since network topology is non-tree, its not friendly to ILP formulation
- The optimization engine is designed using a group migration heuristics
- Similar to stage I, the objective is to minimize the skew cost

Optimization Engine for Stage II

Input: Set S of Component to be sized

1. While (true) {
2. S' ⟵ S
3. While S' is not empty {
4. Find move *i* with max gain g_i
5. e_i = the element sized in move *i*
6. S' ⟵ S' – {e_i}}
7. Find *l* such that cumulated gain G_l is maximized
8. If $G_l > 0$ make the *l* moves on S
9. Else break }

$$g_i = Q_{i-1} - Q_i$$

$$G_l = \sum_{1}^{l} g_i$$

Experimental Setup

- The experiments are performed on ISCAS89 sequential benchmark circuits.
- The circuits are synthesized using SIS and placed in mPL
- The clock tree construction and link insertion is done according to paper G. Venkataraman *et.al* [ICCAD05].

Case	# of Sinks	# of Buffers	# of Links
S9234	135	20	21
S5378	164	25	30
S13207	503	77	69
S15850	566	81	86
S38584	1428	235	50
S35932	1728	286	143

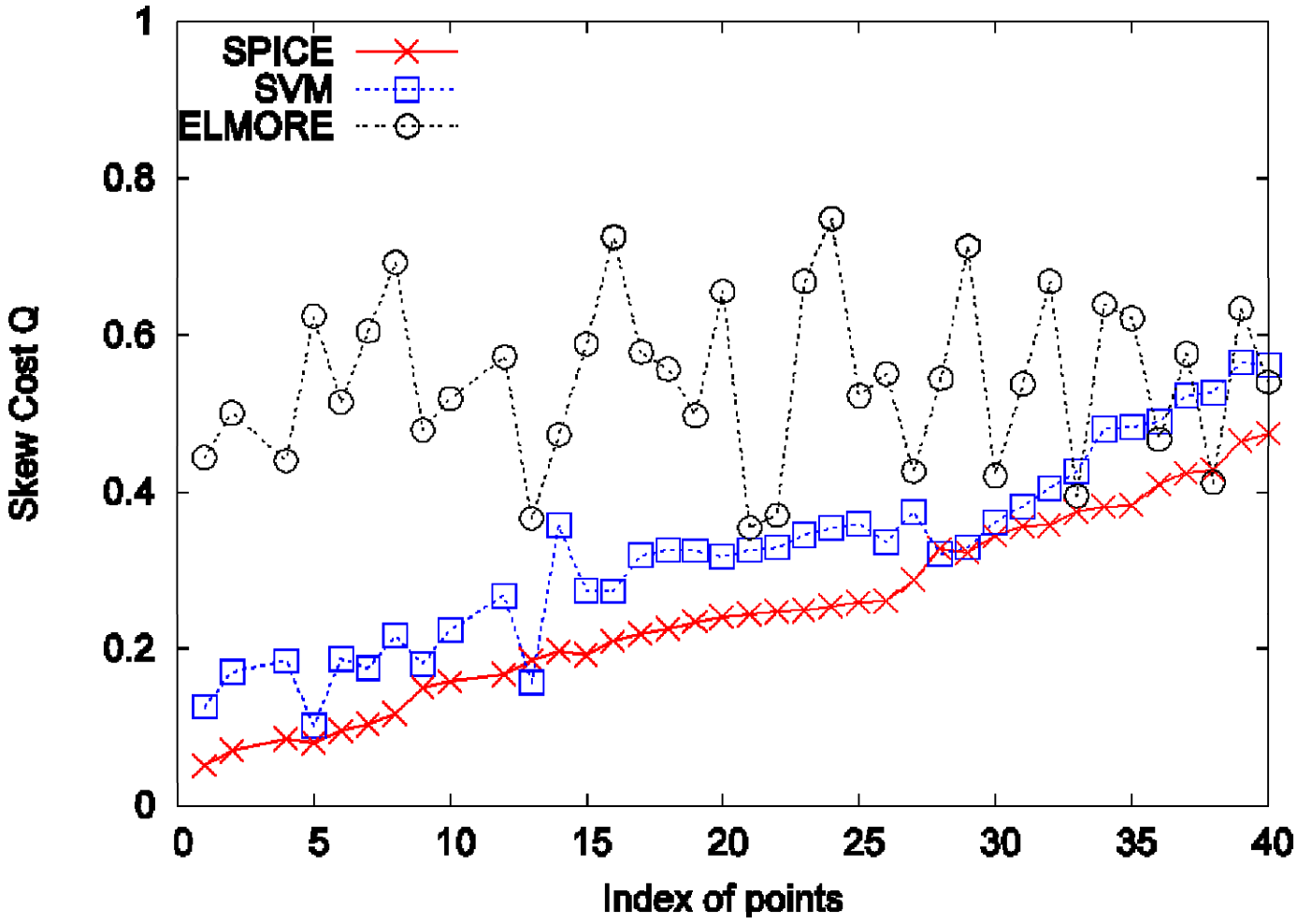
Experimental Setup…*Contd*

Model	Vdd	Buffer Library	Wire Library	K_1	K_2
90nm BPTM	1.0 V	16X, 24X, 32X, 48X	1X, 2X, 3X	10-15	4-5

X: size of minimum width buffer (wire)

- The buffer and wire sizing algorithm is implemented in C.
- The Integer Linear Program is solved using a public domain solver called GLPK [http://www.gnu.org/software/glpk/]
- The binaries for the Support Vector Machine is downloaded from [http://svmlight.joachims.org/]
- The experiments are performed using 2 Dual-Core Intel Xeon Processor of 3.2 Ghz and 8Gb of memory

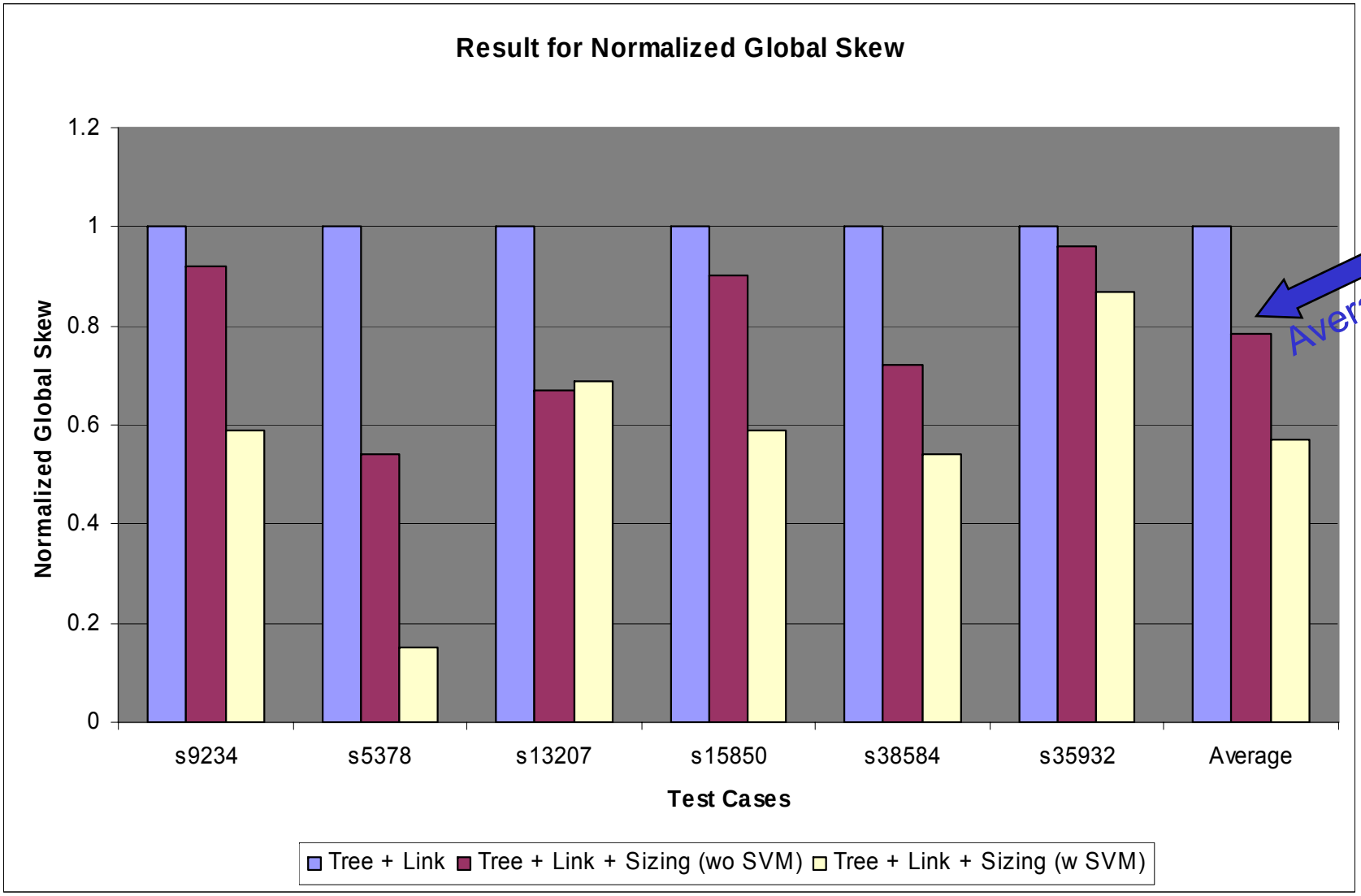
Comparison between SPICE, SVM and Elmore delay



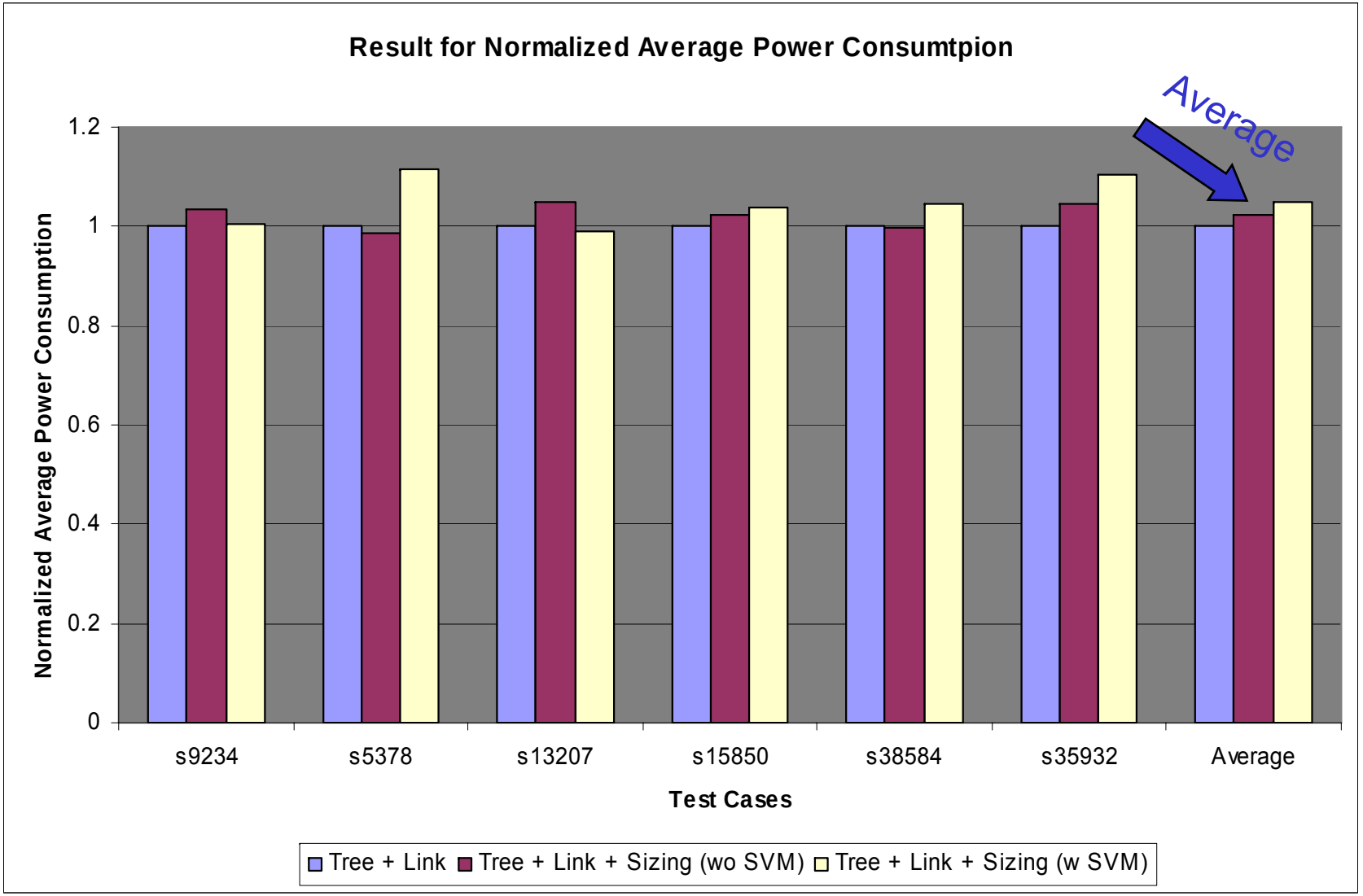
Experimental Results

- The experiments are done in SPICE to compare skew and power
- We compare three approaches for the non-tree clock network
  - Tree+ Link
  - Tree+ Link+ Sizing (wo SVM)
  - Tree+ Link+ Sizing (w SVM)
- Our approach is also suitable for optimizing Clock tree network. So we simulated clock tree for the three approaches
  - Tree
  - Tree + Sizing (wo SVM)
  - Tree + Sizing (w SVM)

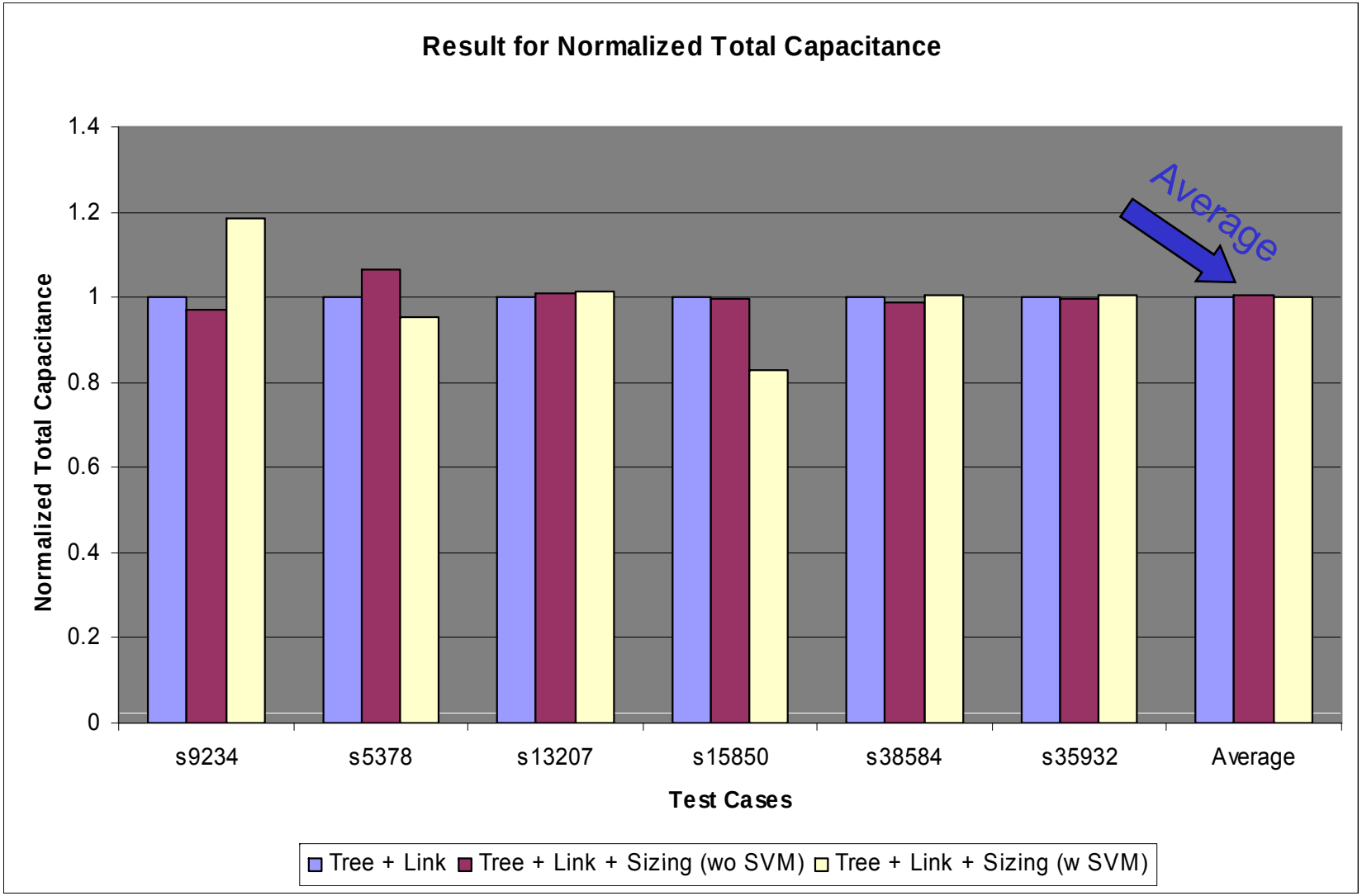
Global Skew for Non-tree Clock Network



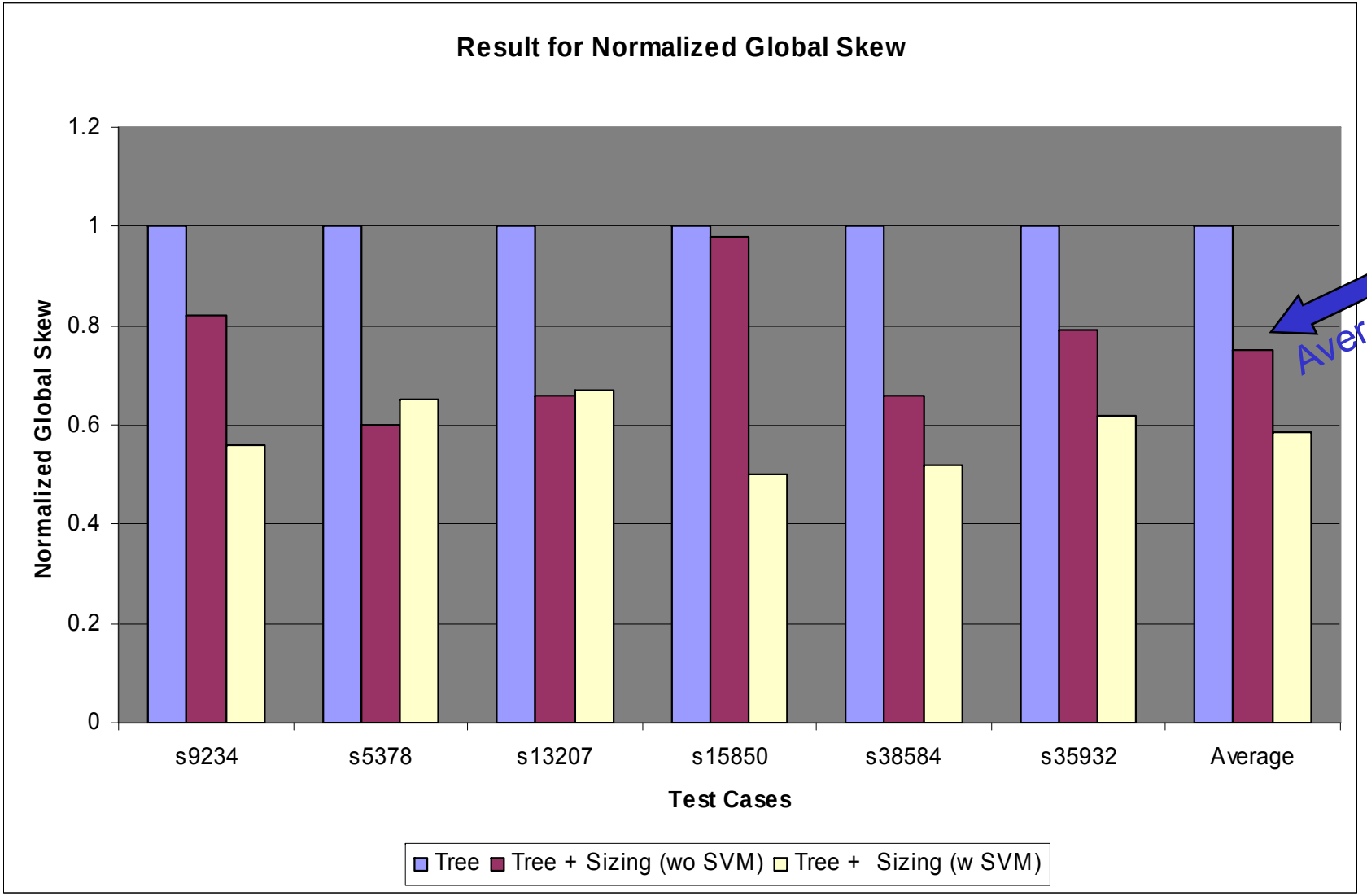
Average Power Consumption for Non-tree Clock Network



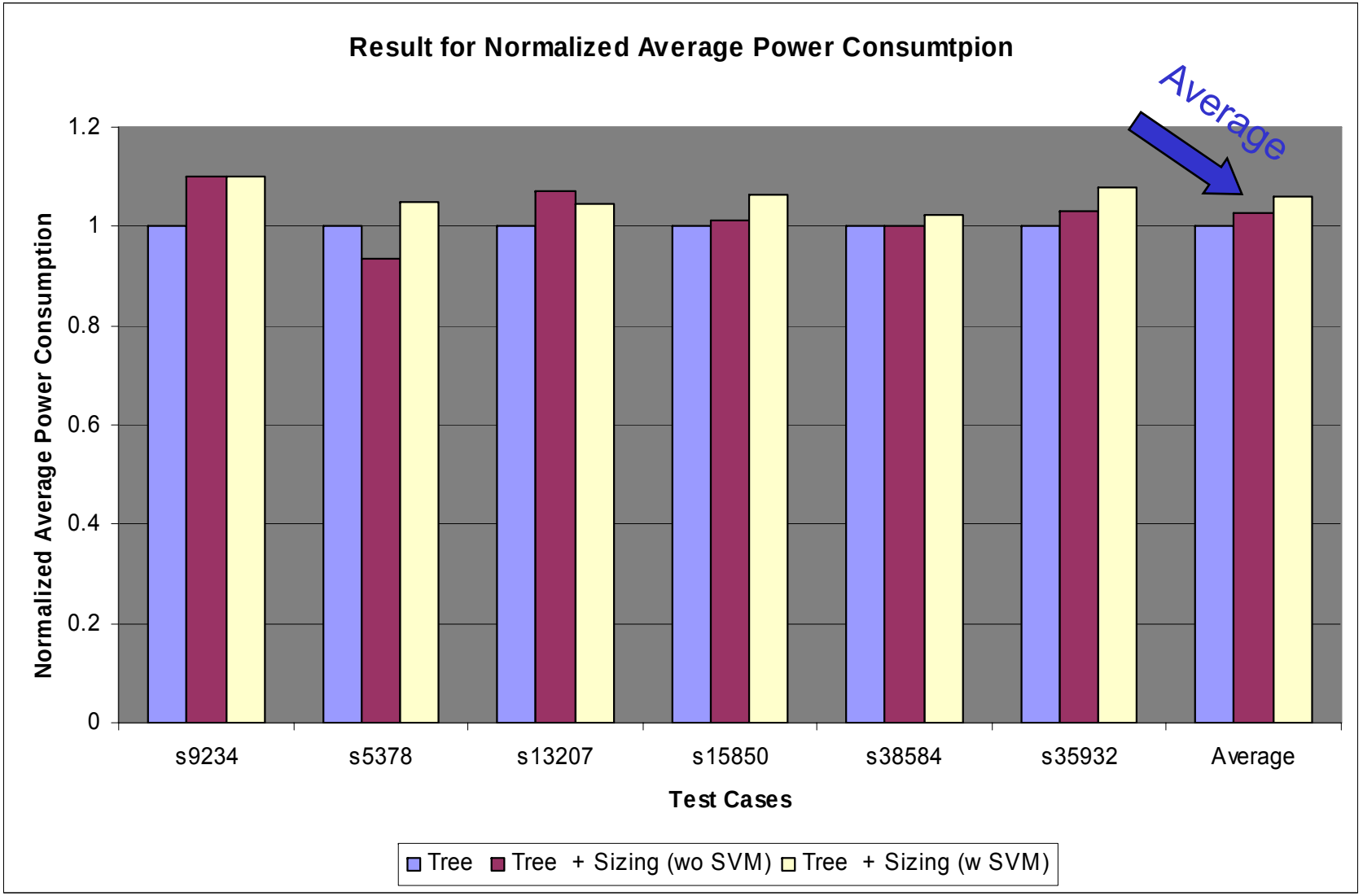
Total Capacitance for Non-tree Clock Network



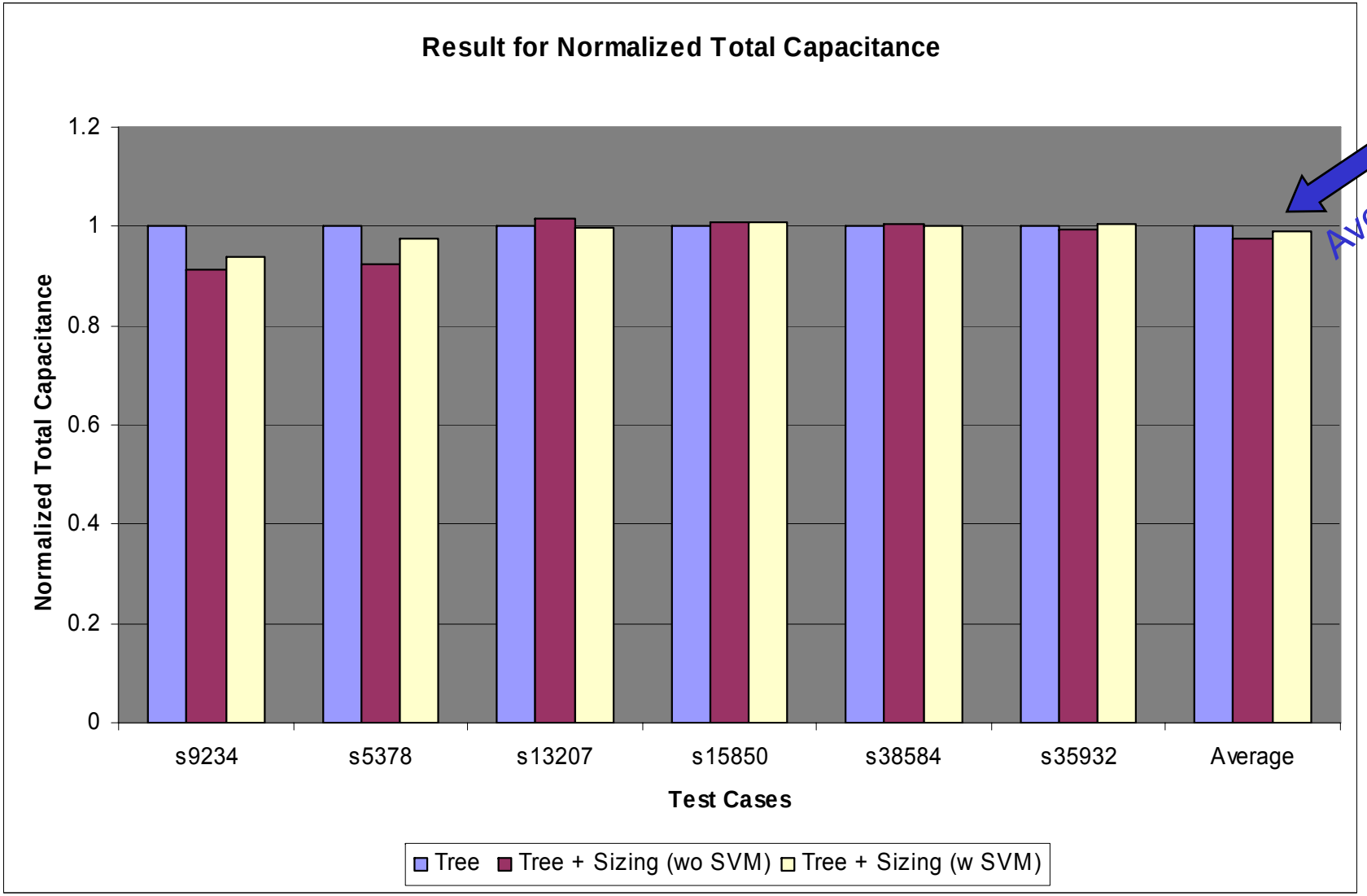
Global Skew for Clock Tree Network



Average Power Consumption for Clock Tree Network



Total Capacitance for Clock Tree Network



Conclusions

- We investigate the buffer and wire sizing for link-based non-tree clock network
- Support Vector Machine is explored
  - to handle complex delay model issues
  - provide guidance in discrete optimization space
- A two stage hybrid optimization approach using an accurate delay is proposed
- SPICE based experimental results indicate significant improvement in skew results
- Our sizing algorithm can also be applied to clock tree network

Questions