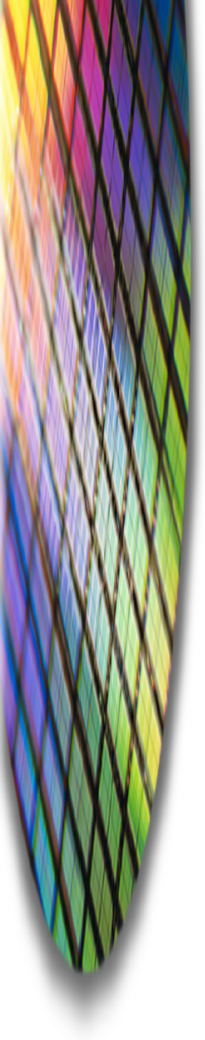


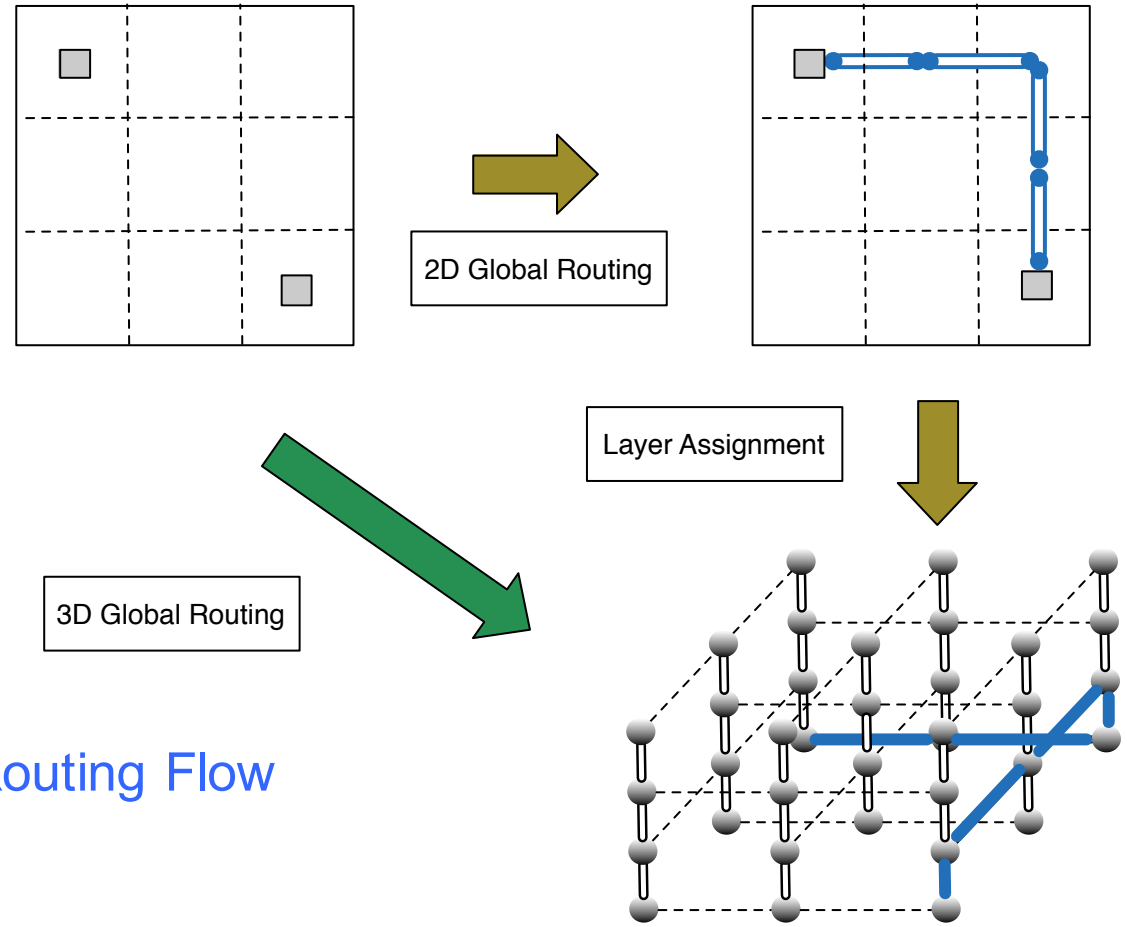
Incremental Layer Assignment for Critical Path Timing

Derong Liu[1], Bei Yu[2], Salim Chowdhury[3], and David Z. Pan[1]

[1]ECE Department, University of Texas at Austin, Austin, TX, USA
[2]CSE Department, Chinese University of Hong Kong, Hong Kong
[3]Oracle Corp., Austin, TX, USA

Introduction

- As VLSI technology scales to deep submicron
  - Interconnect dominates timing issues
  - Global routing – integral part of timing convergence flow



Global Routing Flow

Previous Works on GR and LA

- Example papers on global routing and layer assignment:
  - Timing-driven GR [Liu et al. TCAD'13]
  - Via count and overflow minimization during layer assignment - NVM [Liu+, ASPDAC'11]
  - Delay-driven layer assignment [Yu+, ICCAD'15]

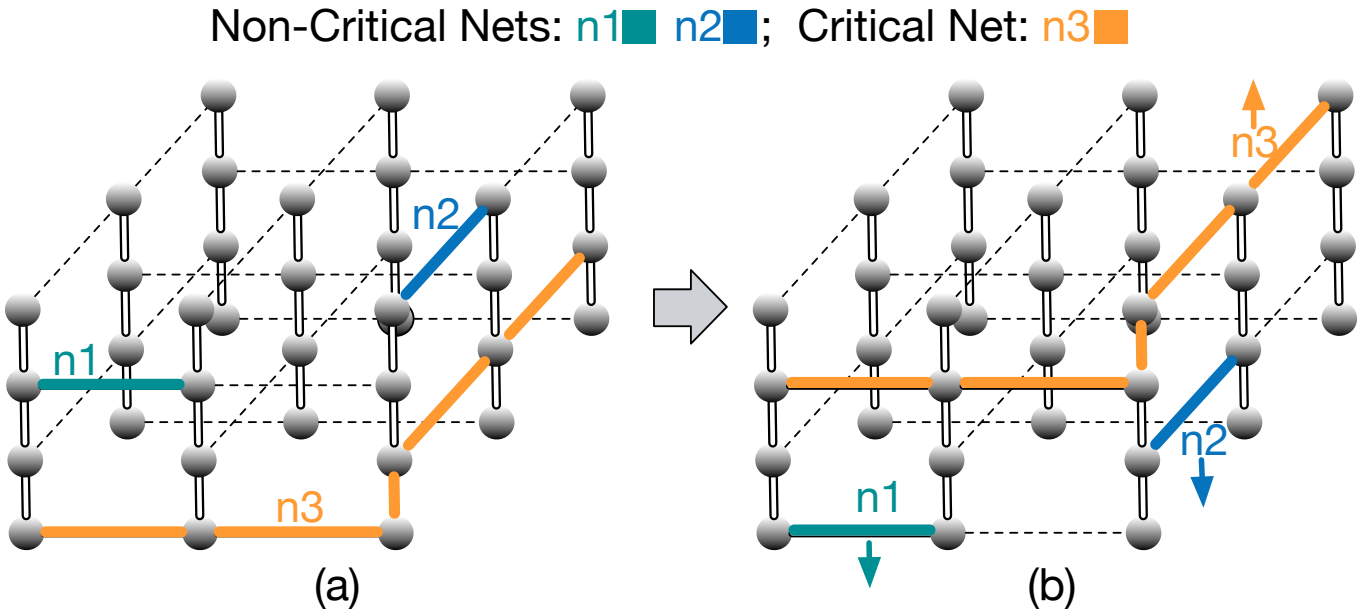
- Limitations of previous layer assignment:
  - Net-by-net method may lead to sub-optimal results
  - Focus on sum of net delays
  - Lack global optimization
  - Linear approximation of via delays influences accuracy.

Contributions of this Work

- A novel incremental layer assignment framework for critical path timing with via delays
- Semidefinite programming (SDP) modeling for better optimal solutions
- Self-adaptive partitioning methodology based on K*K partitions for speed-up

Problem Formulation

- Critical Path Layer Assignment (CPLA)
  - Given a 3-D grid graph, edge and layer information, initial layer assignment solution and set of critical nets
  - Minimize: critical path timing (Elmore delay)
  - Subject to: edge capacity constraints



CPLA Algorithm

- Mathematical Formulation

$$\min \sum_{i \in S(Nc)} \sum_{j}^{L} t_s(i,j) \cdot x_{ij} + \sum_{i,p \in Sx(Nc)} \sum_{j}^{L} \sum_{q}^{L} t_v(i,j,p,q) \cdot y_{ijpq}$$

Segment costs — Via costs

- Constraints:
  - Each segment should be assigned on one and only one layer

$$\sum_{j} x_{ij} = 1, \qquad \forall i \in S(Nc) \quad j \in L$$

  - Edge capacity constraint:

$$\sum_{i \in S(e)} x_{ij} \leq cap_e(j), \qquad \forall e \in E$$

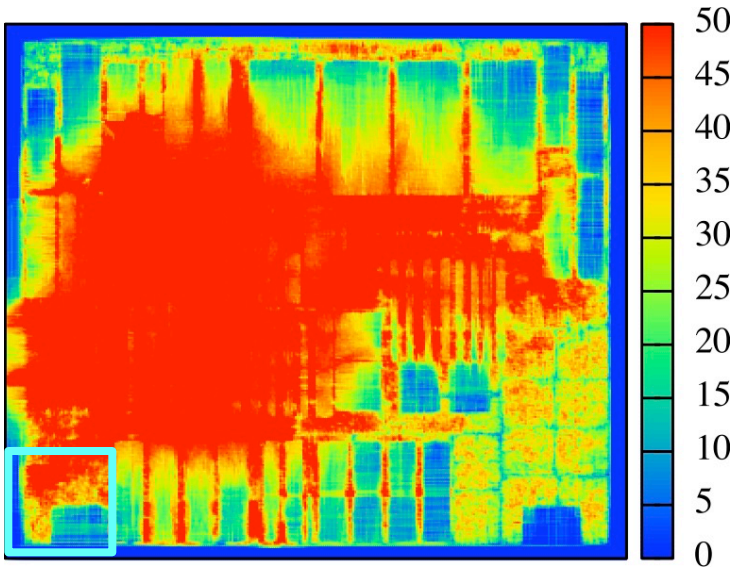
  - Via capacity constraint:

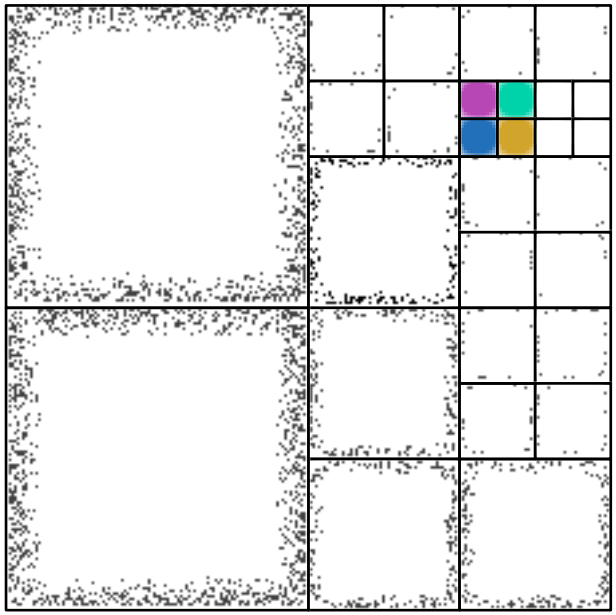
$$\sum_{(i,p) \in Sx(Nc)} y_{ijpq} + n_v(x_{ij} + x_{pq}) \leq cap_g(l), \ \forall l, j < l < q$$

Self-adaptive Quadruple Partition

- K x K division [Yu+, ICCAD'15]
  - Unbalanced computation overhead

- Limit the number of segments in each partition
  - Each thread deals with workload in a well-balanced manner





Semidefinite Programming

$$\min\ (T \cdot X).$$

- Input: parameter matrix T
- Output: variable matrix X

Segment costs

Via costs

$$T = \begin{pmatrix} t_s(i,j) & \dots & t_v(i,j,p,q) \\ \dots & \dots & \dots \\ t_v(i,j,p,q) & \dots & t_s(p,q) \end{pmatrix},$$

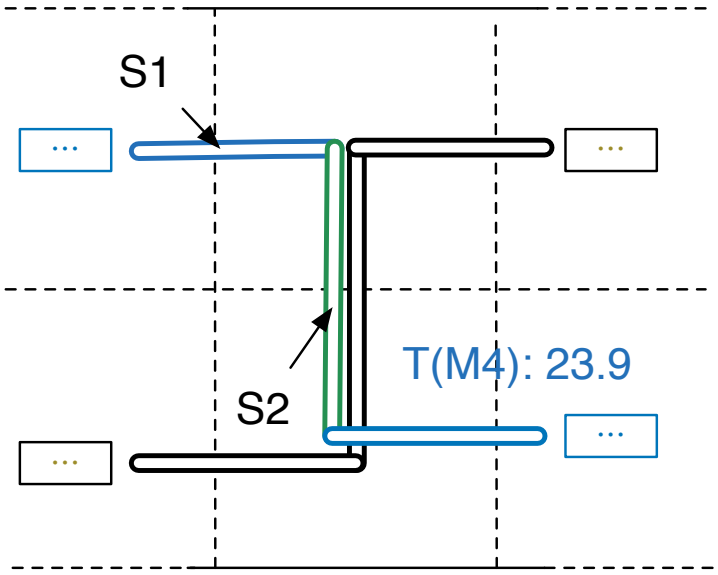
$$X = \begin{pmatrix} x_{ij} & \dots & y_{ijpq} \\ \dots & \dots & \dots \\ y_{ijpq} & \dots & x_{pq} \end{pmatrix}.$$

Example

- Segments S1 and S2
- 4 layers:
  - Layer 1 and 3 for x-dimension
  - Layer 2 and 4 for y-dimension

$$T = \begin{pmatrix} 35.2 & 0 & 5.8 & 6.7 \\ 0 & 15.6 & 2.3 & 3.5 \\ 5.8 & 2.3 & 47.8 & 0 \\ 6.7 & 3.5 & 0 & 23.9 \end{pmatrix}$$

$$X = \begin{pmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.99 & 0.09 & 0.89 \\ 0 & 0.09 & 0.10 & 0 \\ 0 & 0.89 & 0 & 0.90 \end{pmatrix}$$



- Post mapping strategy provides integer solutions

Experimental Results

- Implemented the framework in C++
- ILP solver -- GUROBI
- SDP solver – CSDP
- Parallel implementation – OpenMP

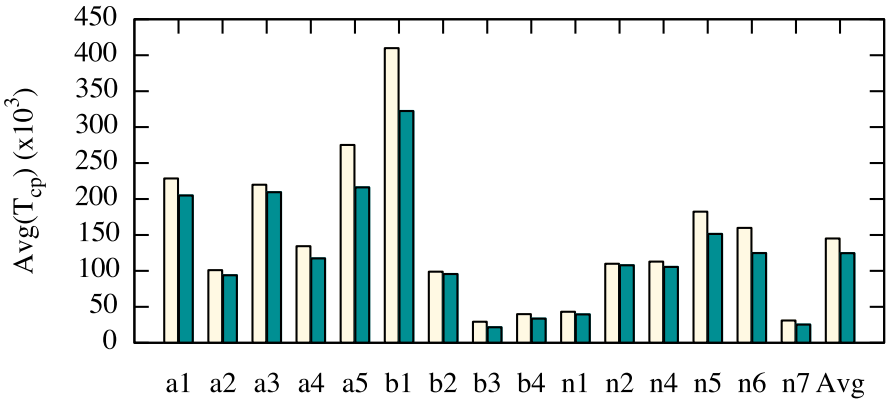
- Evaluation on ISPD'08 global routing benchmarks
- Performance Metrics
  - Average Critical Path Timing
  - Worst Critical Path Timing
  - # of Via capacity violation
  - # of Via

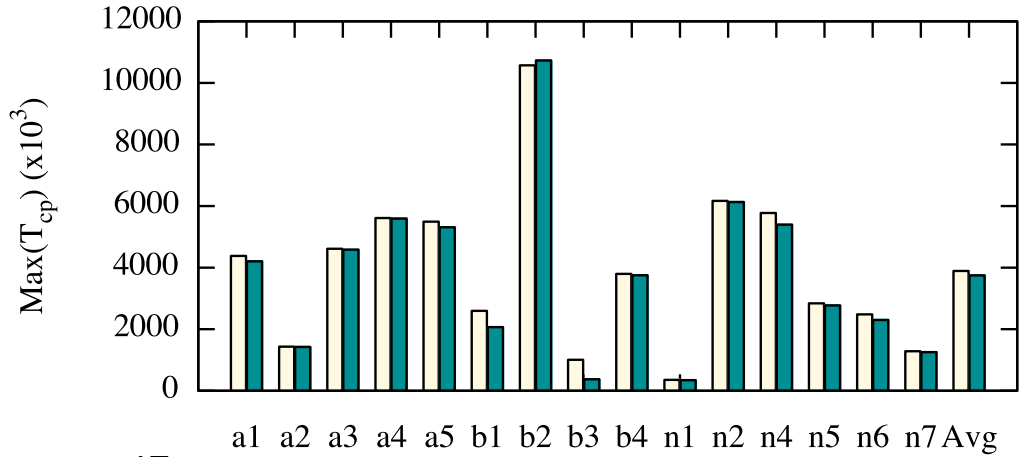
Evaluation on ISPD'08 Benchmarks

- Initial global routing input:
  - Generated by **NCTU-GR 2.0** [Liu et al. TCAD'13]
- Initial layer assignment:
  - From **NVM** [Liu et al. ASPDAC'11]
  - Targeting via number and overflow minimization
- Wire resistance and capacitance values obtained from industry settings
- Release 0.5% critical and non-critical nets
- Compared with **TILA** [Yu et al. ICCAD'15]

Delay Comparison Results

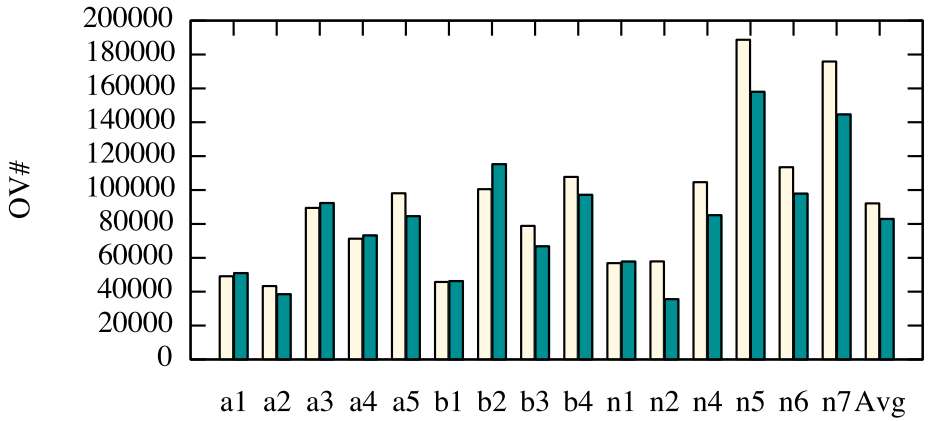
- ISPD'08 Global Routing Benchmarks
  - 14% improvement in $Avg(T_{cp})$
  - 4% improvement in $Max(T_{cp})$

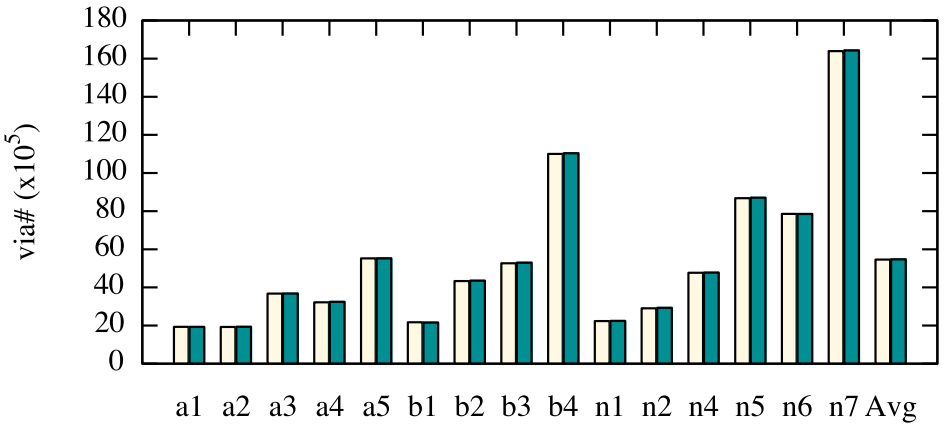




Via Comparison Results

- ISPD'08 Global Routing Benchmarks
  - Via Overflow (#OV) decreases by 10%
  - Similar number of vias





Conclusion

- Propose Incremental Layer Assignment for Critical Path Timing (**CPLA**) algorithm
  - Self-adaptive partition provides balanced workload for multiple threads and potential speed-up
  - Semidefinite programming (SDP) relaxation
  - Post mapping satisfies constraints
- CPLA suitable for future heterogeneous layer structures

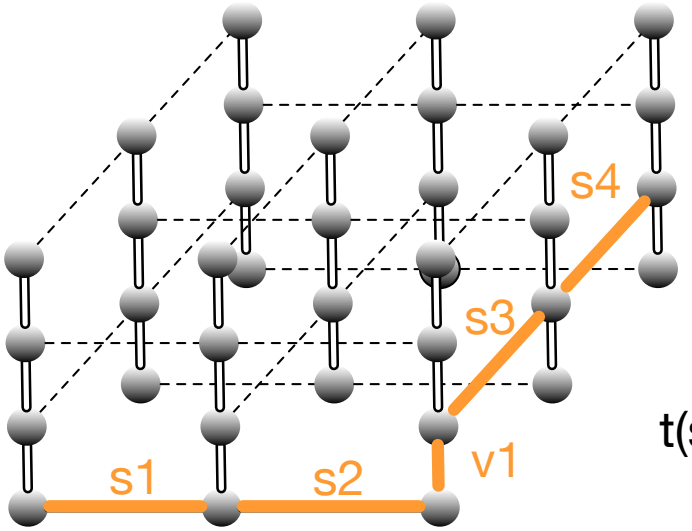
Thanks Q&A

Overview

- Introduction
- Problem Formulation
- Algorithms
- Experimental Results
- Conclusion

Model Description

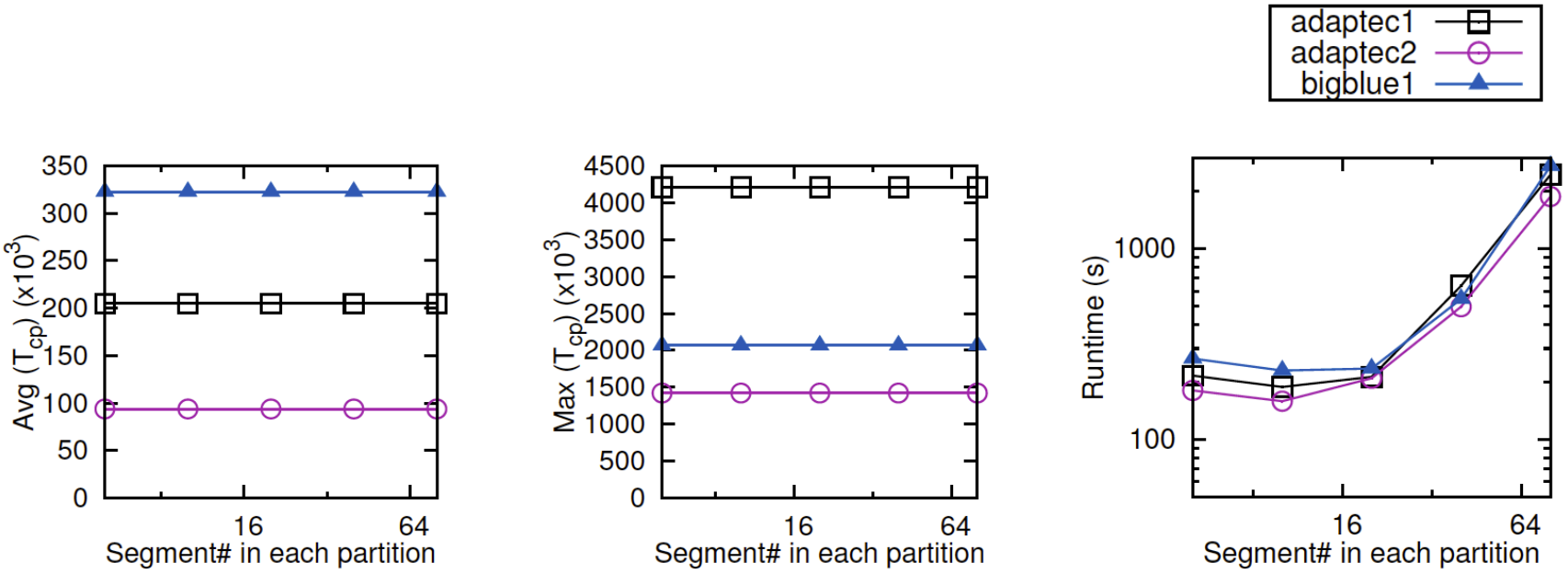
- Elmore timing model:
  - Consider both segment delay and via delay



Maximum Path Delay:
t(s1) + t(s2) + t(s3) + t(s4) + t(v1)

Partition Size Impact

- Different number of segments in each partition



Impact on $Avg(T_{cp})$;

Impact on $Max(T_{cp})$;

Impact on runtime

Mapping Algorithm

- Provide integer solutions
  - For each edge with critical nets: