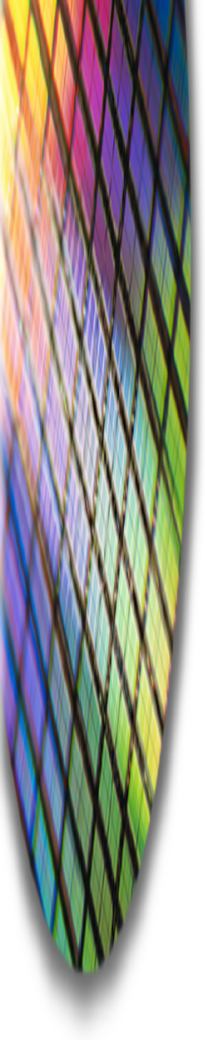


TILA: Timing-Driven Incremental Layer Assignment

Bei Yu[1,2], **Derong Liu[1]**, 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
[3]Oracle Corp., Austin, TX, USA

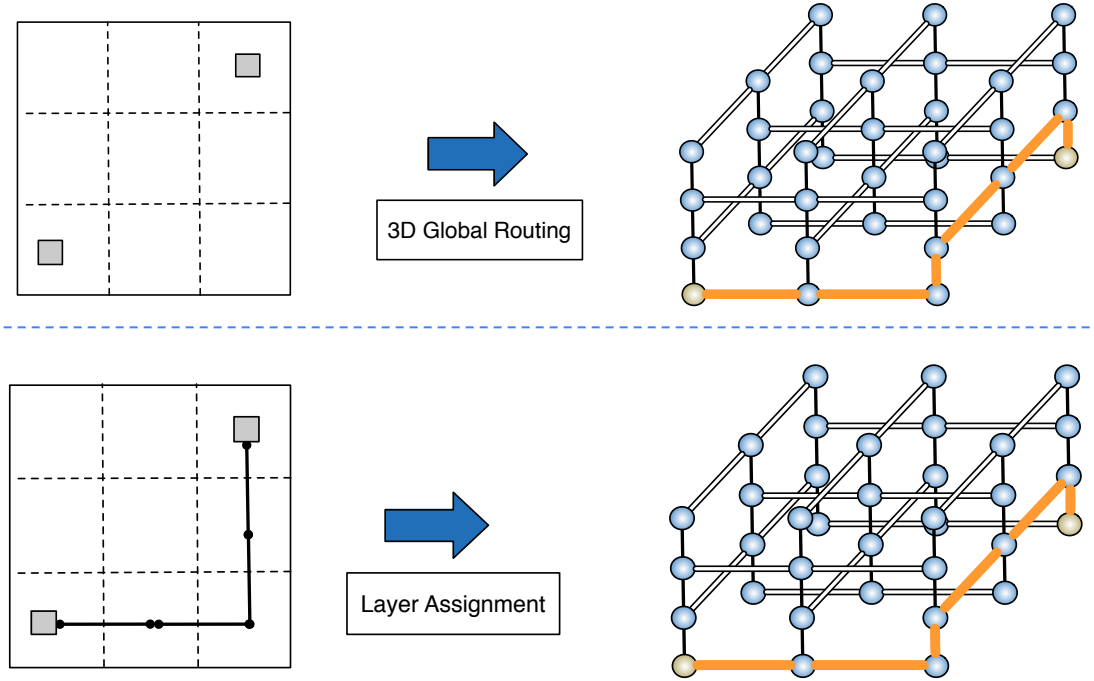
Overview

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

Introduction

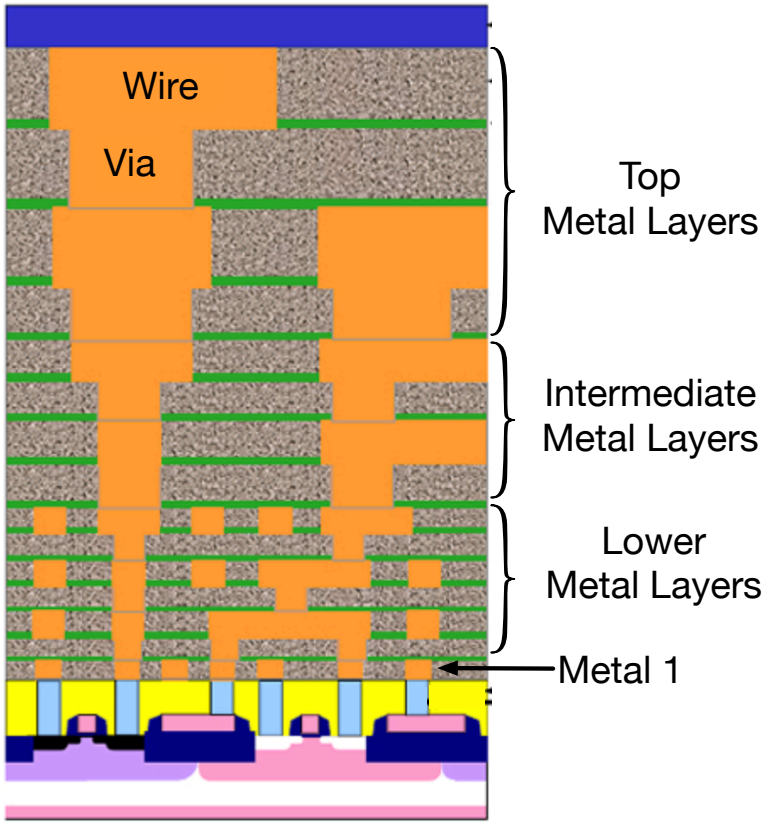
- VLSI technology scales to deep submicron
  - Interconnect delay
- Interconnect synthesis
  - Timing-driven routing
  - Global routing -- part of a timing convergence flow

Global Routing Flow



Introduction

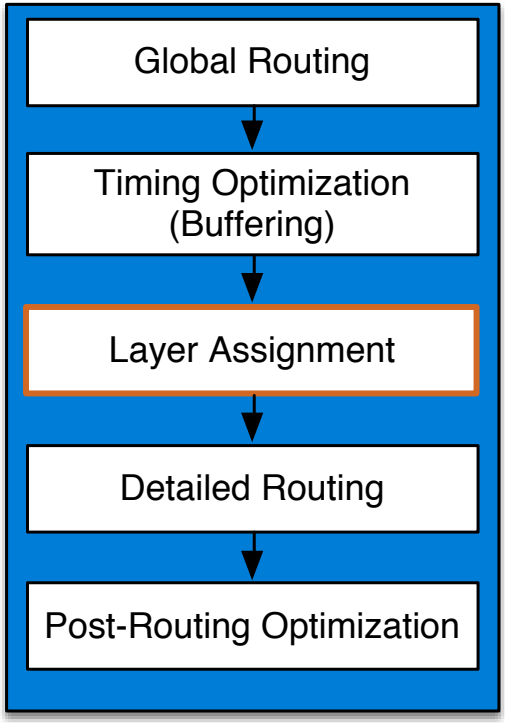
- **Layer assignment:**
  - Assign segments to metal layers
  - Impossible to assign all segments on higher layers



Wire [Hsu et al. ICCAD'14]			Via	
Layer	C	R	Layer	R
M1	1.14	23.26	$v_{1,2}$	25.9
M2	1.05	19.30	$v_{2,3}$	16.7
M3	1.05	23.26	$v_{3,4}$	16.7
M4	0.95	5.58	$v_{4,5}$	16.7
M5	1.05	3.26	$v_{5,6}$	5.9
M6	1.05	3.26	$v_{6,7}$	5.9
M7	1.05	3.26	$v_{7,8}$	5.9
M8	1.00	3.26	$v_{8,9}$	1.0
M9	1.05	1.00	$v_{9,10}$	1.0
M10	1.00	1.00	-	-

Previous Works on GR and LA

- Many papers on global routing and layer assignment, e.g.
  - Routability-driven GR [Cho+, TCAD'07]
  - Timing-driven GR [Liu et al. TCAD'13]
  - Via count and overflow minimization during layer assignment - NVM [Liu+, ASPDAC'11]
  - Delay-driven layer assignment [Ao+, ISPD'13]
- Limitations of previous layer assignment:
  - Most focus on via minimization
  - Via delays are often ignored
  - Net-by-net method may lead to local optimality

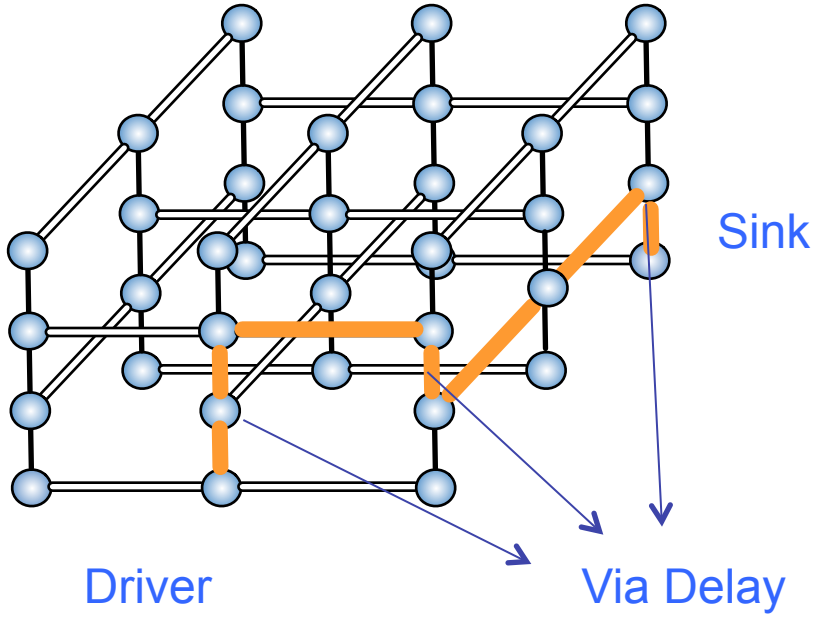


Contributions of this Work

- The first timing-driven incremental layer assignment (TILA) that integrates via delay
- Solve multiple nets simultaneously
- Incremental approach to provide fast turn-around-time
- Lagrangian relaxation based optimization to improve total wire and via delay via min-cost flow iteratively
- Multi-processing of K∗K partitions for speed-up
- Effectiveness demonstrated by ISPD'08 and industry benchmarks

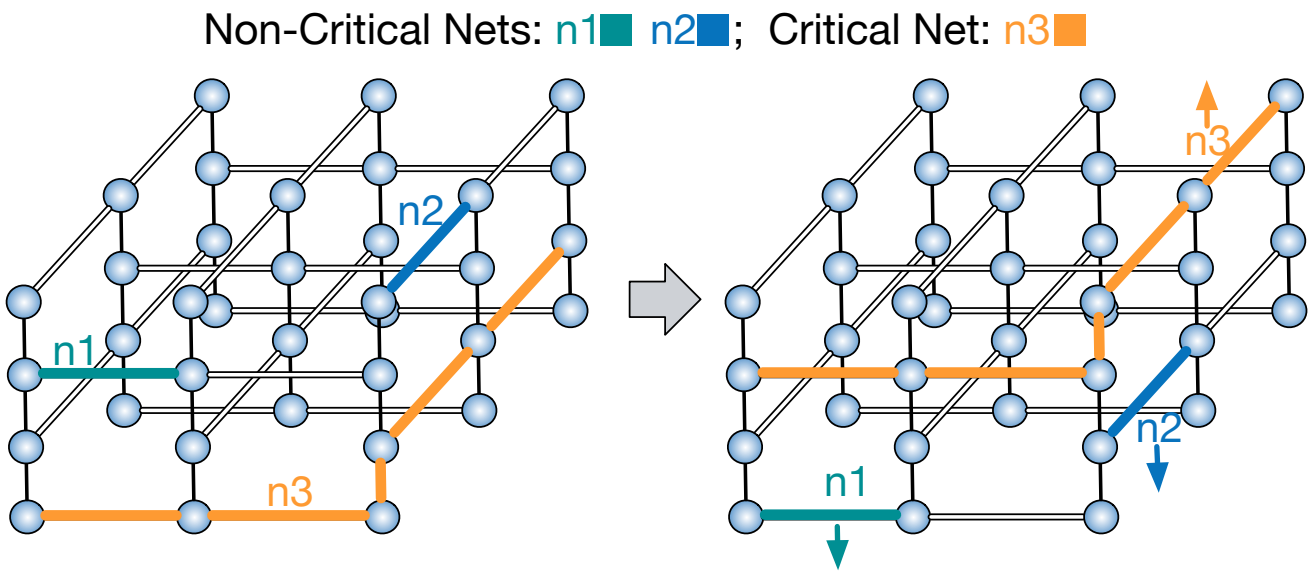
Model Description

- Timing model:
  - Elmore Delay (C_{down}, R)
  - Consider both segment delay and via delay



Problem Formulation

- Timing-driven Incremental Layer Assignment (TILA)
  - Given initial layer assignment solution and critical ratio α
  - Minimize: sum of segment and via delays of selected nets
  - Subject to: via capacity and edge capacity constraints



One Example

TILA Algorithm

- Mathematical Formulation

$$\min \sum_{i=1}^{S}\sum_{j=1}^{L} d_e(i,j)\cdot a_{ij} + \sum_{(i,p)\in E_x}\sum_{j=1}^{L}\sum_{q=1}^{L}\sum_{k=j}^{q-1} d_v(i,p,k)\cdot a_{ij}\cdot a_{pq}$$

Segment Delay — Via Delay

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

$$\sum_j a_{ij} = 1, \qquad \forall i \in [1,S]$$

  - Edge capacity constraint:

$$\sum_{s_i \in S_e(i)} \sum_j a_{ij} \le c_e(i), \qquad \forall e \in E$$

  - Via capacity constraint:

$$\sum_{(i,p)\in E_x} a_{ij}\cdot a_{pq} \le c_v(k), \quad \forall k, j<k<p$$

TILA Algorithm

- Lagrangian Relaxation Subproblem (LRS)

$$\min \sum_{i=1}^{S}\sum_{j=1}^{L} d_e(i,j)\cdot a_{ij} + \sum_{(i,p)\in E_x}\sum_{j=1}^{L}\sum_{q=1}^{L}\sum_{k=j}^{q-1} d_v(i,p,k)\cdot a_{ij}\cdot a_{pq}$$
$$+ \sum_{(i,p)\in E_x} \lambda_{ij,pq}(a_{ij}\cdot a_{pq} - c_v(k))$$

Integrate via capacity constraint with Lagrangian Multipliers (LMs)

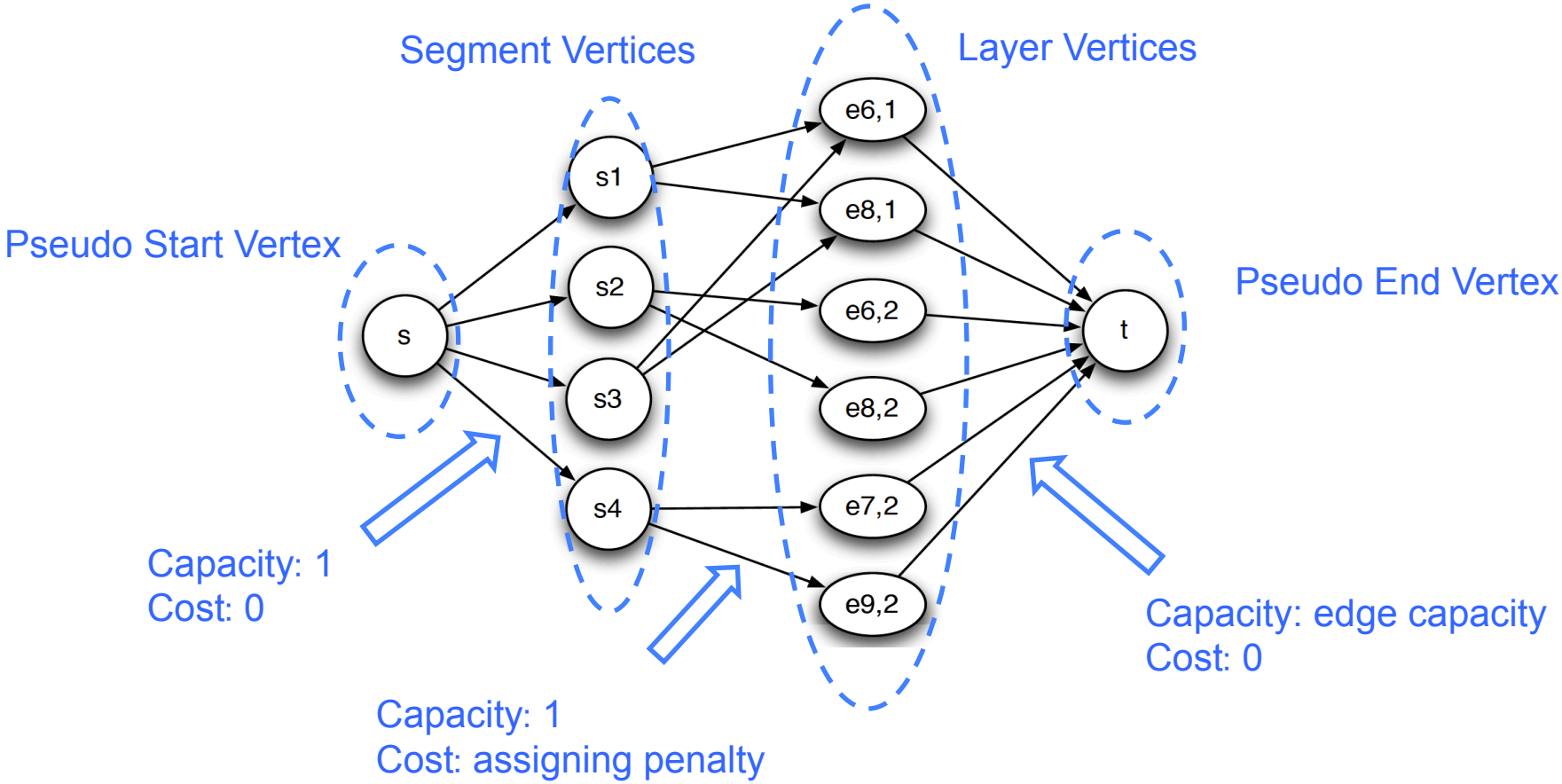
  - Solve the LRS iteratively
- Linearize the quadratic term approximately:

$$a_{ij}\cdot a_{pq} \approx a'_{pq}\cdot a_{ij} + a'_{ij}\cdot a_{pq}$$

  - Based on the values in previous iteration
- Solve the LRS through a min-cost network flow model
  - Satisfy the edge capacity constraint
  - Guarantee one segment on one layer

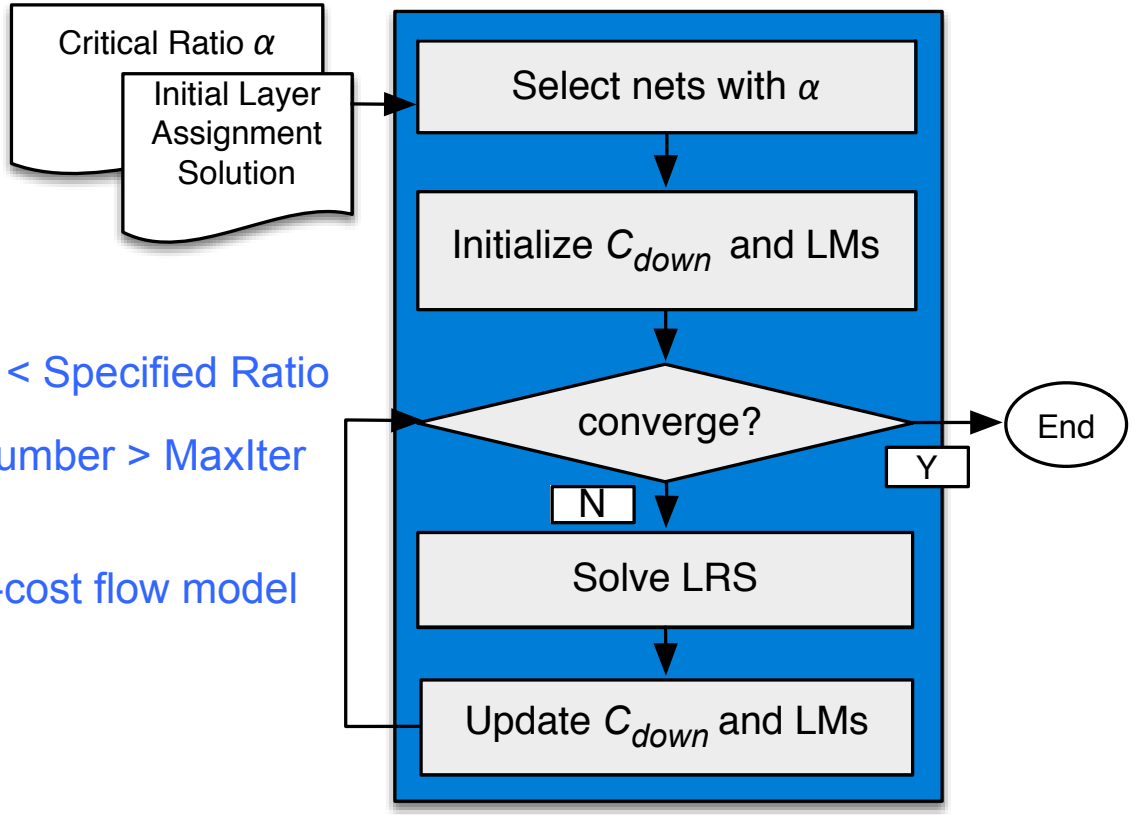
TILA Algorithms

- Min-cost Flow Model
  - Inherent uni-modular property to ensure integer solutions
  - Directed Graph G (V,E)



TILA Algorithm

- Algorithm Flow



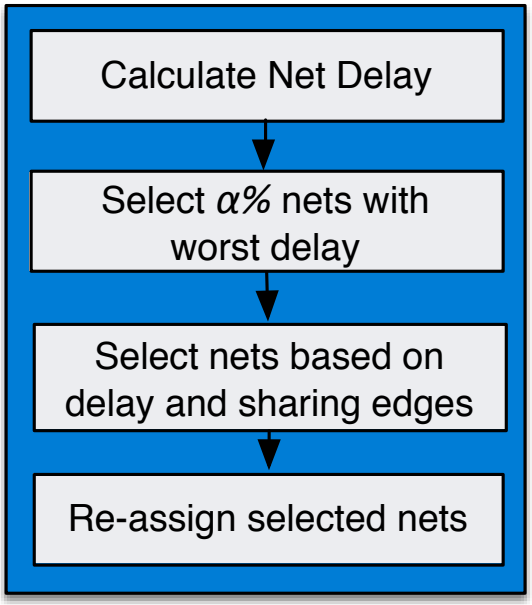
Improvement < Specified Ratio

Or Iteration number > MaxIter

Min-cost flow model

Incremental Approach

- Critical & Non-critical Net Selection
  - Most Critical nets: improve the timing
  - Most non-critical nets: release more high layers resources



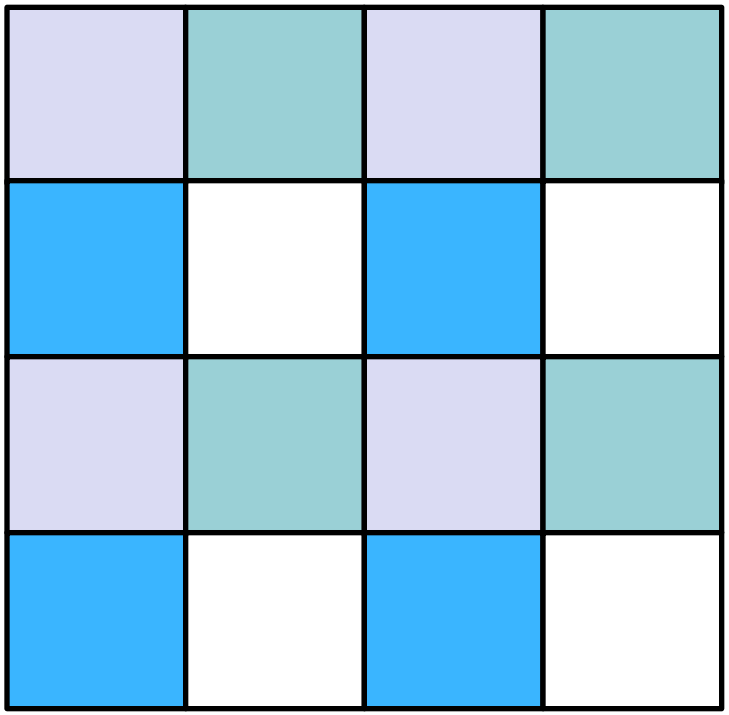
Nets Selection Flow

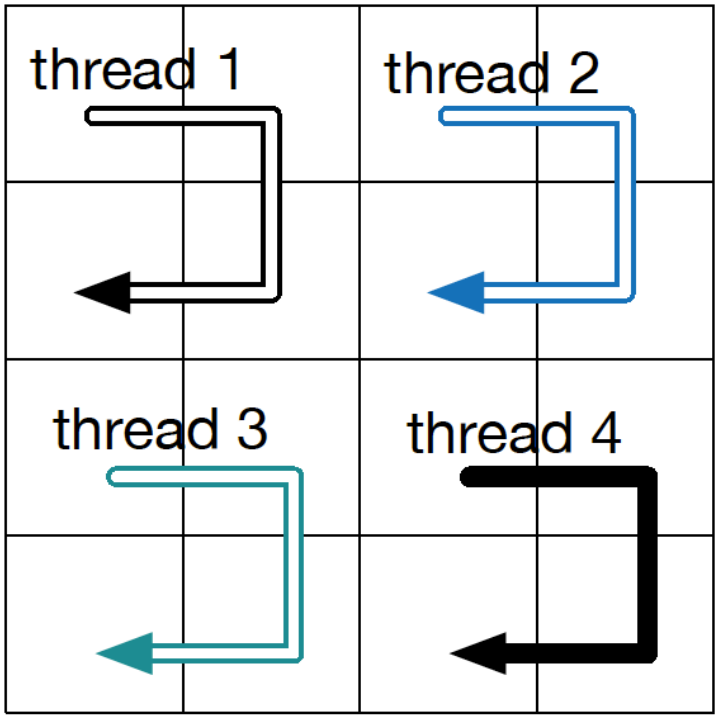
Most critical nets selection

Most non-critical nets selection

Speed-up Techniques

- Parallel Scheme
  - Divide grid model into *K* x *K* partitions
  - Recent results by peer threads can be considered





4*4 partitions

Experimental Results

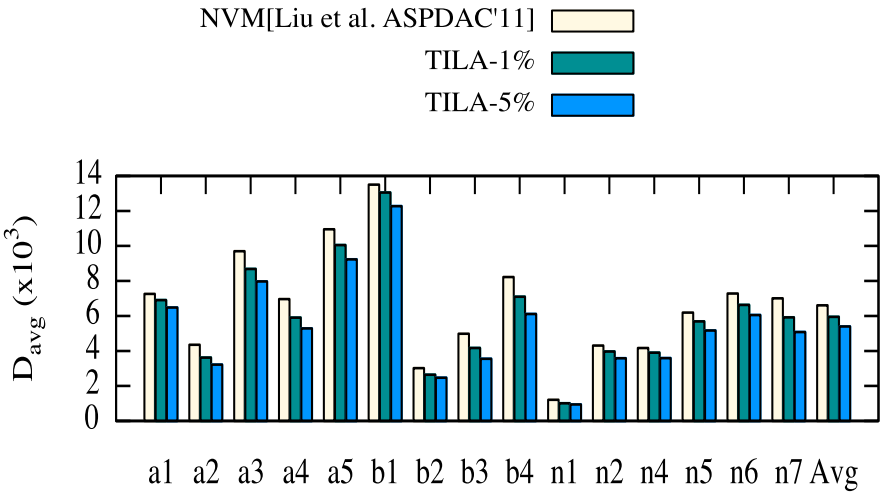
- Implemented the framework in C++
- Tested on Linux machine with eight 3.3GHz CPUs
- Min-cost flow solver
  - LEMON open source graph library
- Parallel computation with OpenMP
  - Default thread number as 6 and K set as 6
- Evaluation on both academia and industrial benchmarks
- Performance Metrics
  - Average Delay
  - Maximum Delay
  - Via capacity violation#
  - Via#

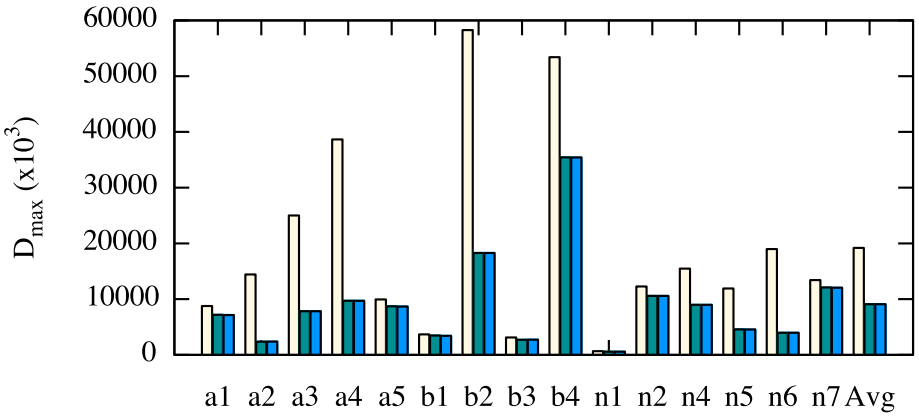
Evaluation on ISPD'08 Benchmarks

- 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 [Hsu et al. ICCAD'14]
- Via resistance and capacitance normalized from industry
- Release 1% and 5% critical and non-critical nets

Delay Comparison Results

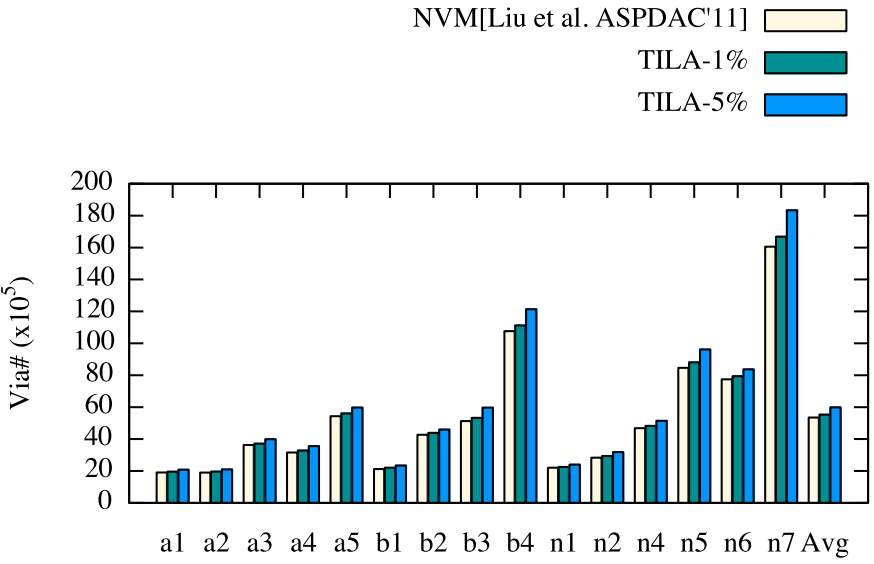
- ISPD'08 Global Routing Benchmarks
- TILA-1%:
  - 53% improvement by D_{max} and 10% improvement by D_{avg}
- TILA-5%:
  - 53% improvement by D_{max} and 18% improvement by D_{avg}

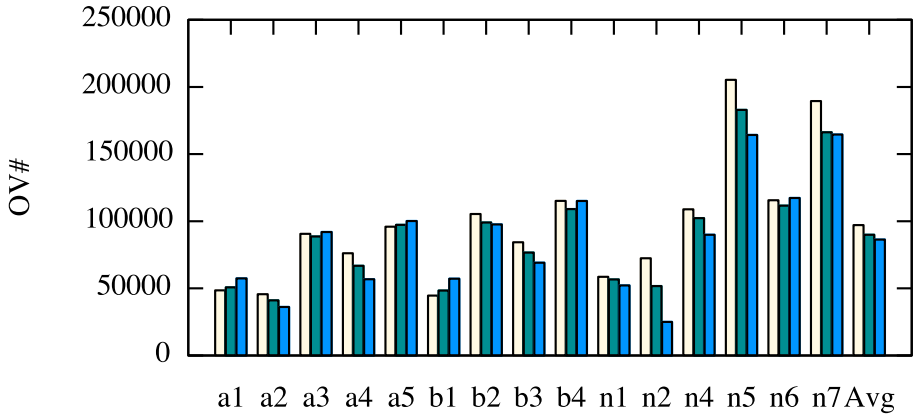




Via Comparison Results

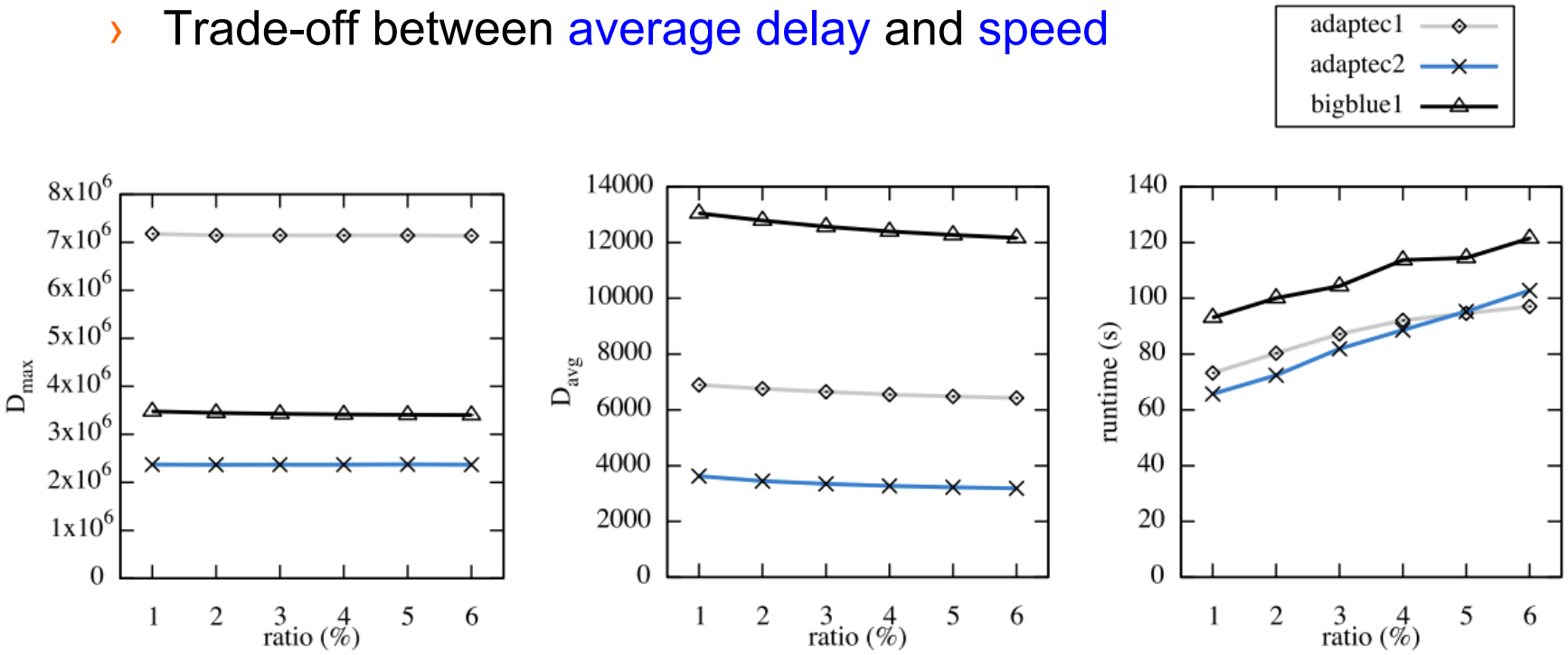
- ISPD'08 Global Routing benchmarks
- TILA-1%
  - OV# decreases by 7% and Via# increases by 3%
- TILA-5%
  - OV# decreases by 11% and Via# increases by 12%





Experimental Results

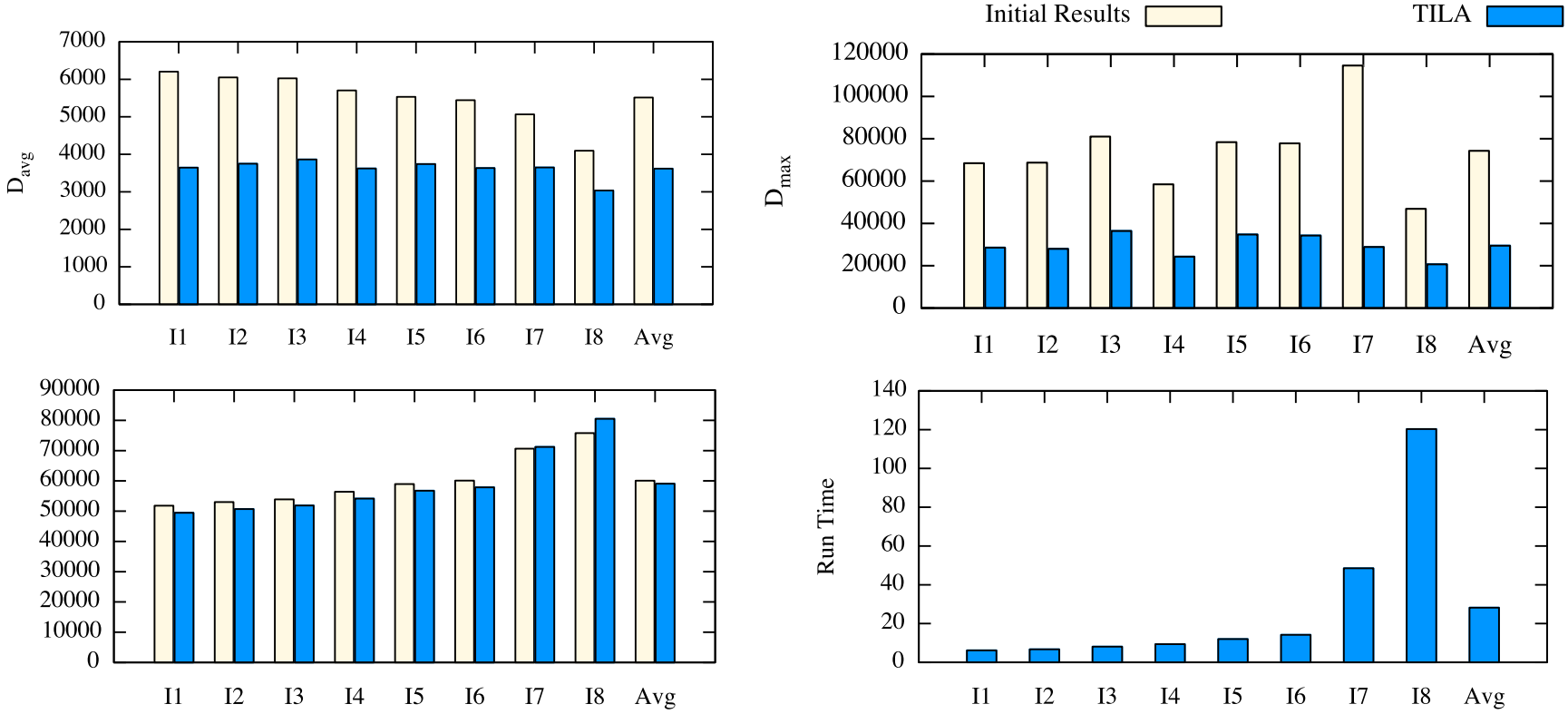
- Impact of different critical/non-critical ratio
  - Releasing 1% is enough for maximum delay
  - Trade-off between average delay and speed



D_{max}: max delay; D_{avg}: average delay; run time

Industrial Benchmarks Results

- Industry tool to generate initial routing solution
- Use industry resistance and capacitance



OV#: overflow number; D_{avg}: average delay; D_{max}: max delay; via#: total via number

Conclusion

- We proposed a new Timing-driven Incremental Layer Assignment (TILA) algorithm
  - Select a subset of critical and non-critical nets
  - Lagrangian relaxation based global optimization
  - Min-cost network flow to solve iteratively
  - Multi-threading
- TILA can work smoothly with any global router and adapt easily to future heterogeneous layer structures
- New research needed to shed light on “classical” EDA problems

Thanks Q&A