An Enhanced Global Router With Consideration of General Layer Directives

Tsung-Hsien Lee[1], Yen-Jung Chang[1], and Ting-Chi Wang[2]

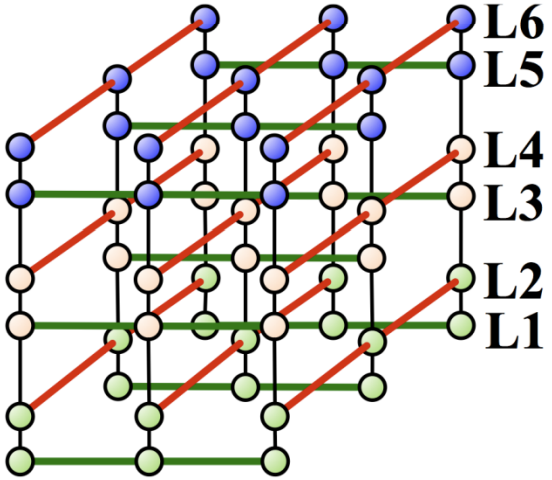
[1]Department of Electrical and Computer Engineering, University of Texas at Austin
[2]Department of Computer Science, National Tsing Hua University, Taiwan

Agenda

- Background
- Problem formulation
- Previous work: GLADE
- Our router
- Experimental results
- Conclusion

Global Routing

- Global routing determines tile-to-tile routes of nets
- Conventional Metrics
  - Total overflow (TOF)
  - Total wirelength (TWL)



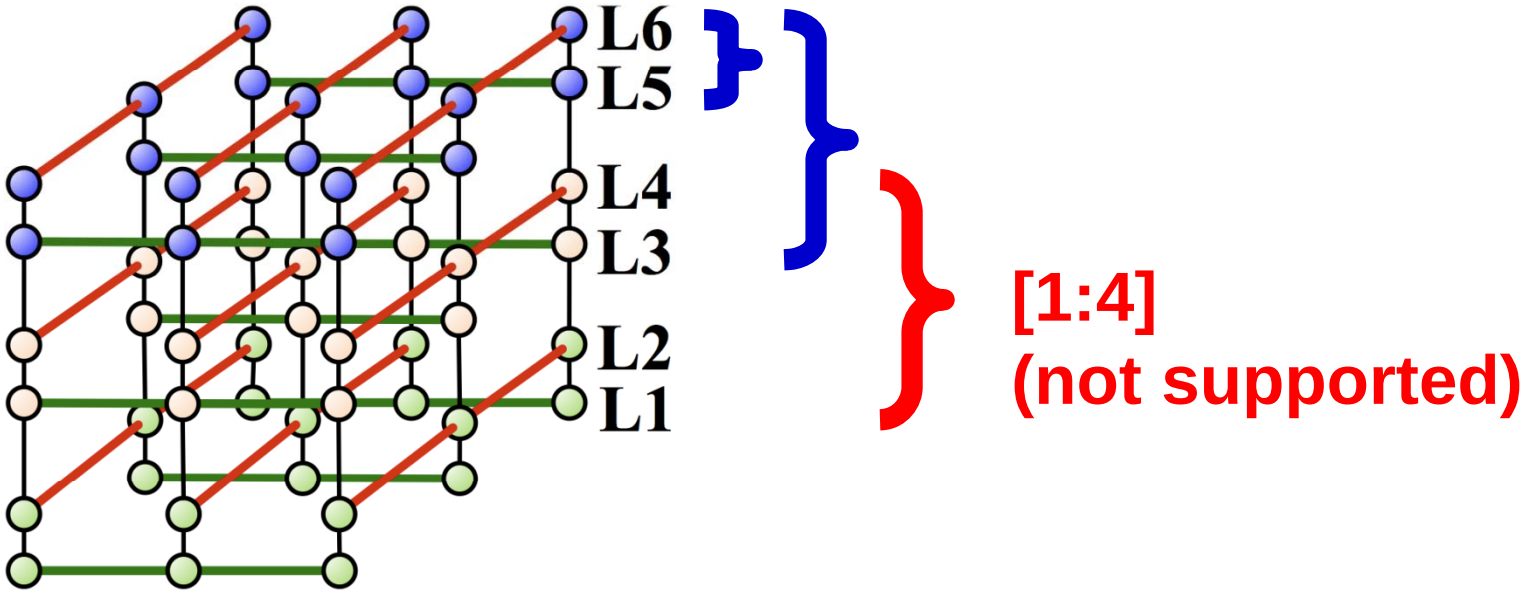
3D Grid Graph for Global Routing

ICCAD 2009 Benchmarks (1/2)

- Produced by making some modifications to ISPD 2008 benchmarks
- Specifying layer directives for a subset of nets (LD nets)
  - Layer directive: a range of consecutive layers on which the net should be routed

ICCAD 2009 Benchmarks (2/2)

- Different LD types whose layer ranges have proper subset relations
- Do not support arbitrary layer ranges



Problem Formulation

- Input: a multi-layer global routing instance with a subset of nets associated with **general layer directives**
  - The two ends of a layer range can be any metal layers
- Output: a global routing solution that minimizes
  - total LD violation as well as TOF
    - A LD net passing through an edge on a non-preferred layer causes one unit of LD violation on the edge
  - TWL

Previous Work: GLADE [ICCAD10]

- Handling ICCAD 2009 benchmarks and hence only targeting a restricted set of layer ranges
- Extending NTHU-Route 2.0 [TCAD10] by performing
  - Pseudo layer assignment during 2D routing
  - LD-aware layer assignment

GLADE: Pseudo Layer Assignment

- Exploited during 2D global routing
- Predicting the amount of LD violations that may occur after actual layer assignment, subject to no overflow increase
- Calculating virtual capacity (VC) and virtual demand (VD) which are also used to define edge costs for LD nets during the iterative ripup-and-reroute process

Illustration of VC (1/4)

- 3D edges e'_1, e'_2, e'_3, e'_4 are projected to a 2D edge e
- Three LD types: t1, t2, t3

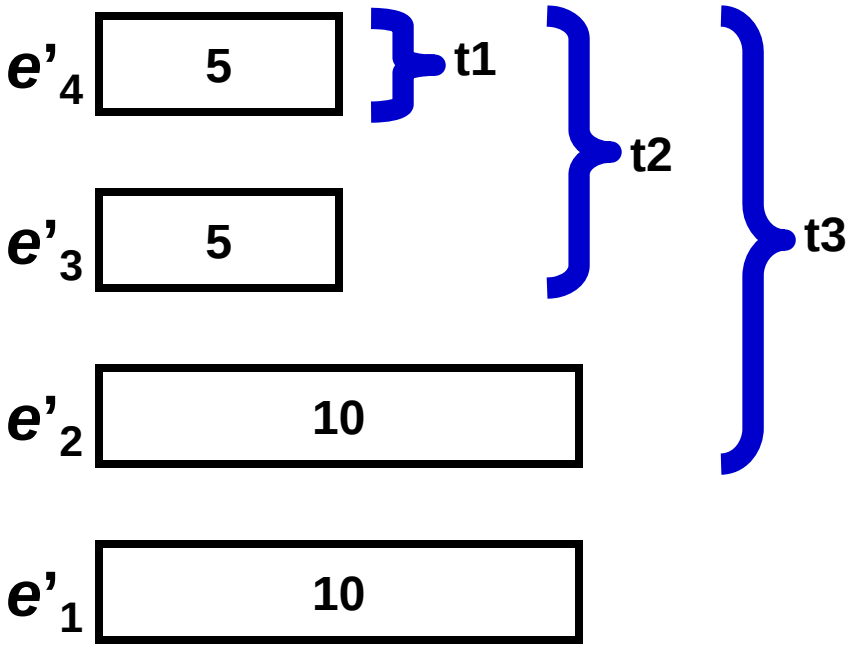


Illustration of VC (2/4)

- $vc_e(t1) = 5$

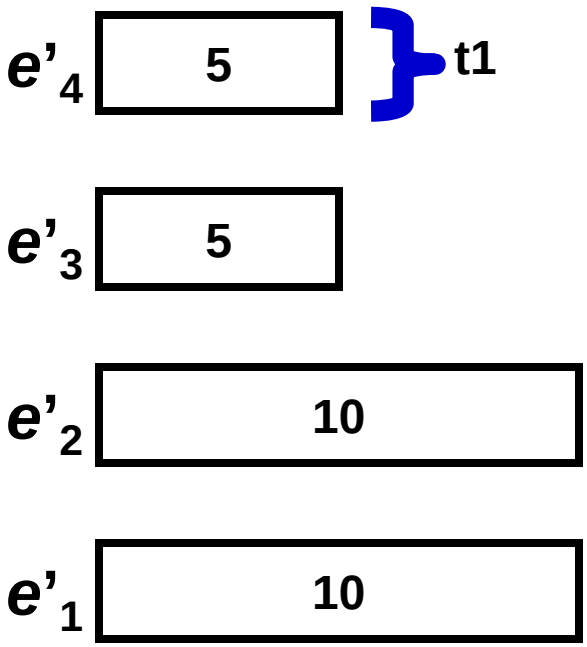


Illustration of VC (3/4)

- $vc_e(t2) = 5 + 5 = 10$

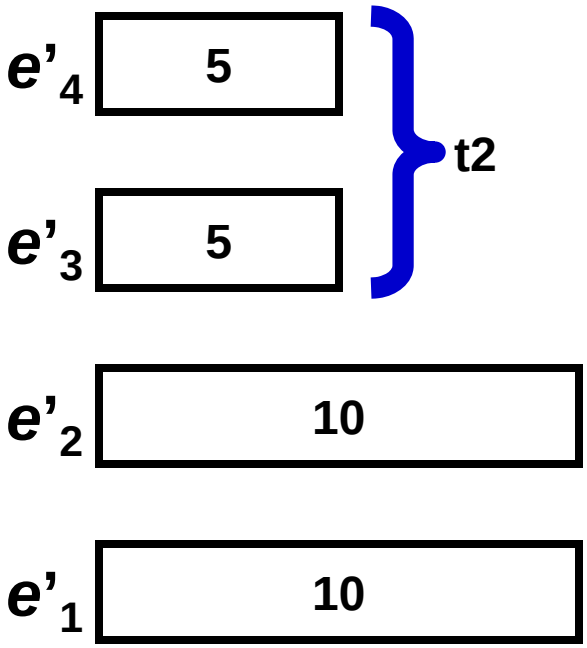


Illustration of VC (4/4)

- $vc_e(t3) = 5 + 5 + 10 = 20$

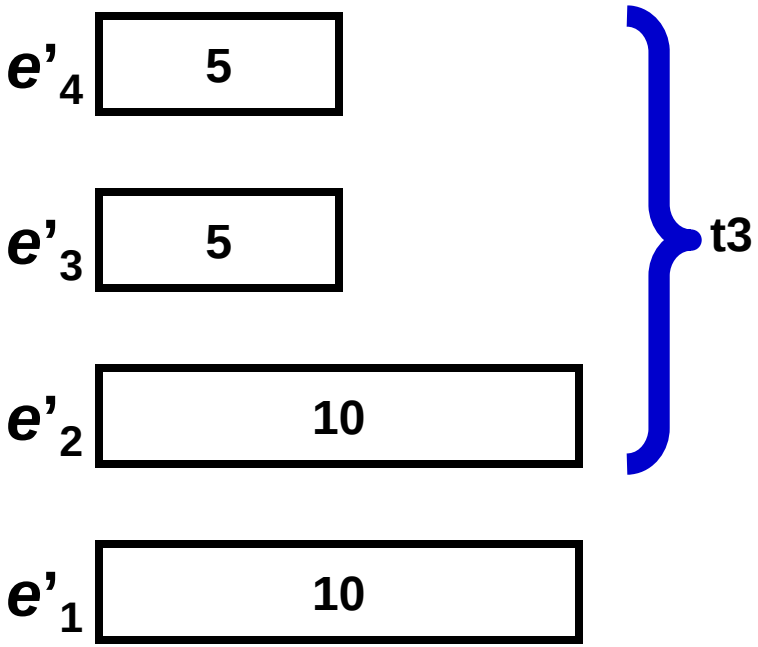


Illustration of VD (1/3)

- $vd_e(t1) = 4$

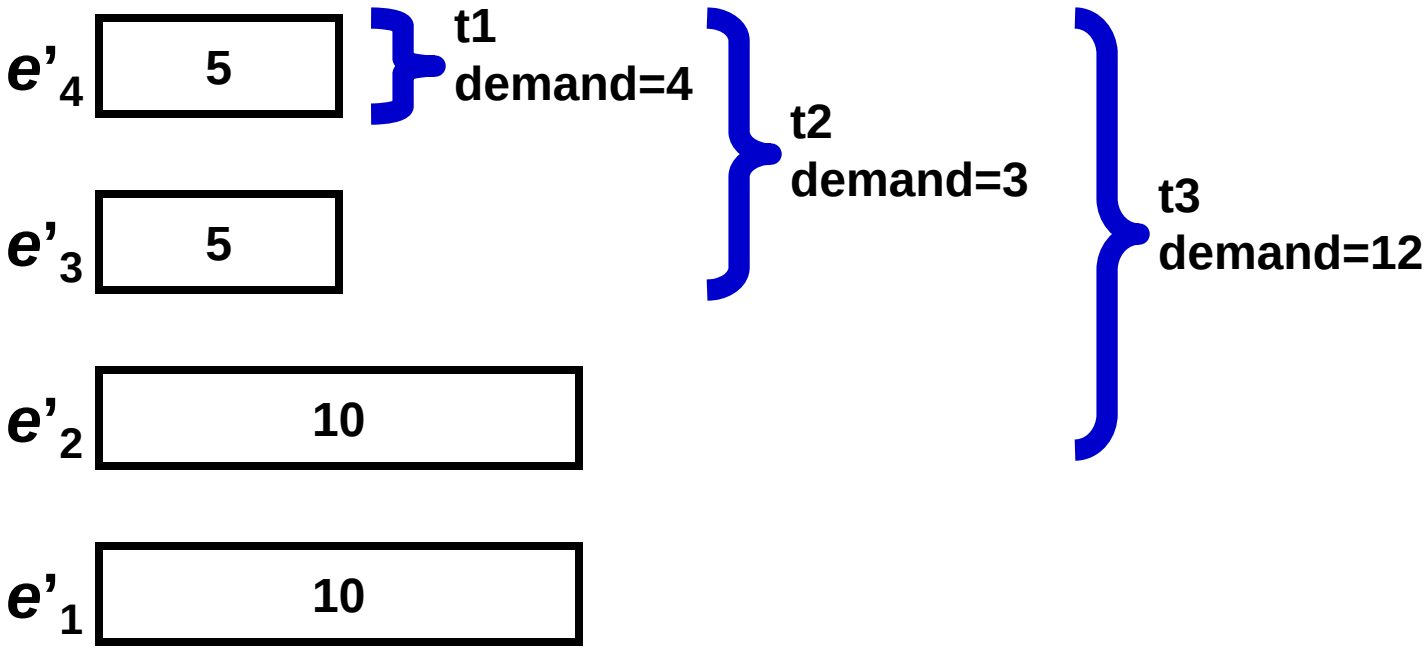


Illustration of VD (2/3)

- $vd_e(t1) = 4$
- $vd_e(t2) = 4 + 3 = 7$

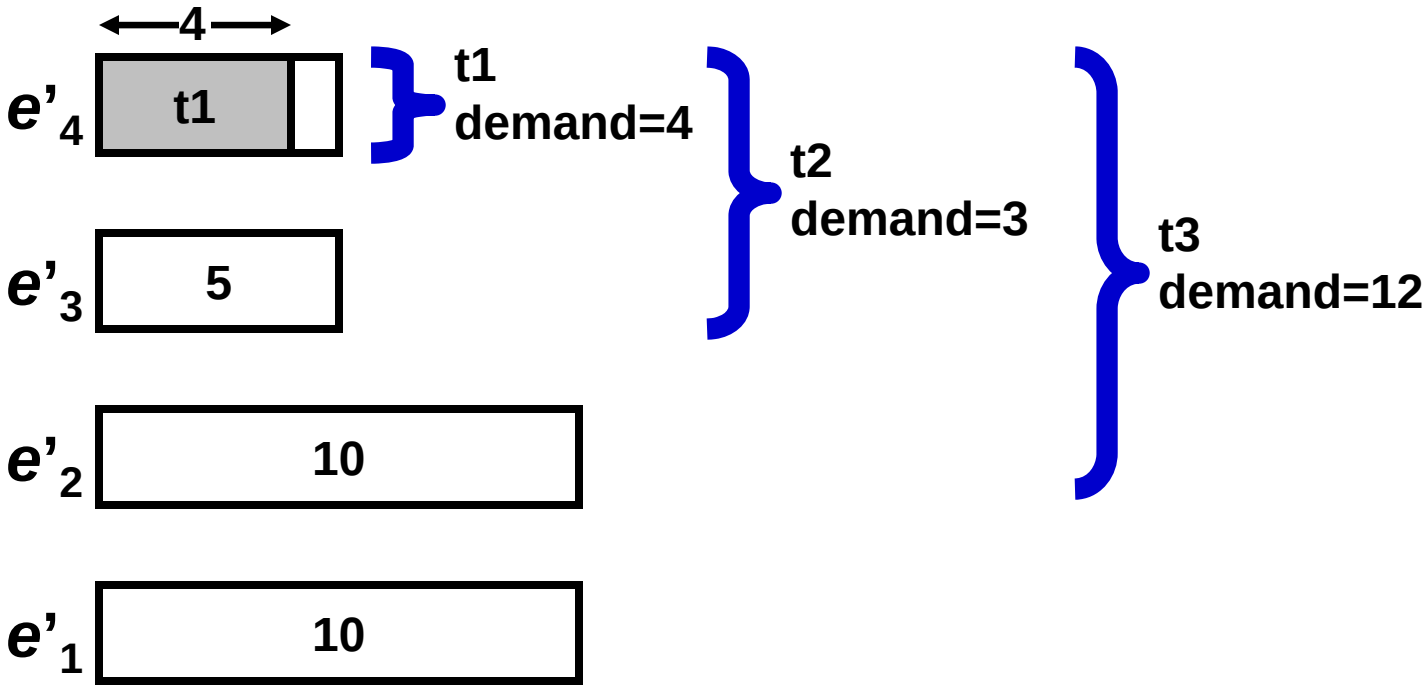
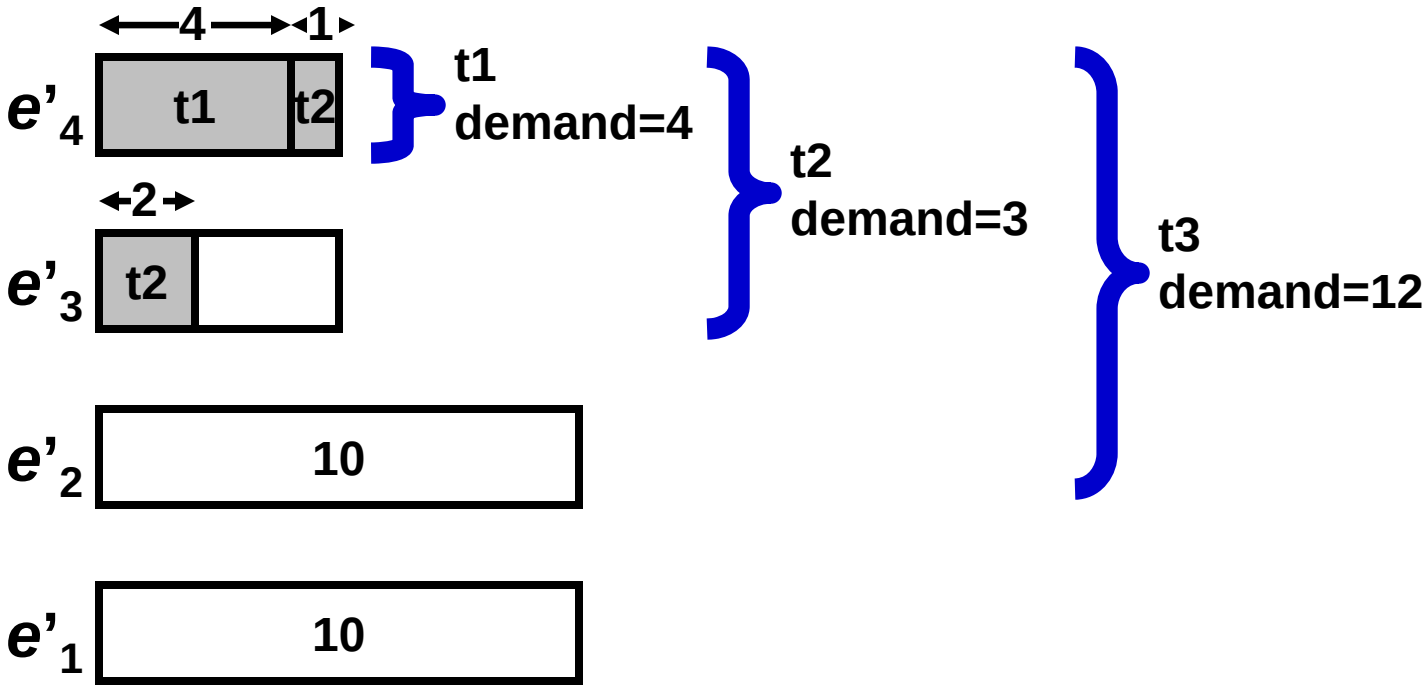


Illustration of VD (3/3)

- $vd_e(t1) = 4$
- $vd_e(t2) = 4 + 3 = 7$
- $vd_e(t3) = (4 + 1 + 2) + 12 = 19$



LD Overflow (LDOF)

- $LDOF_e(t) = \max(vd_e(t) - vc_e(t), 0)$
  - How many LD nets of type t that pass through e cannot be assigned to their preferred layers without causing additional overflow
- $LDOF_e = \Sigma_t\ LDOF_e(t)$
- Total $LDOF = \Sigma_e\ LDOF_e$
- At each ripup-and-reroute iteration, GLADE tries to minimize TOF and total LDOF

GLADE: Layer Assignment

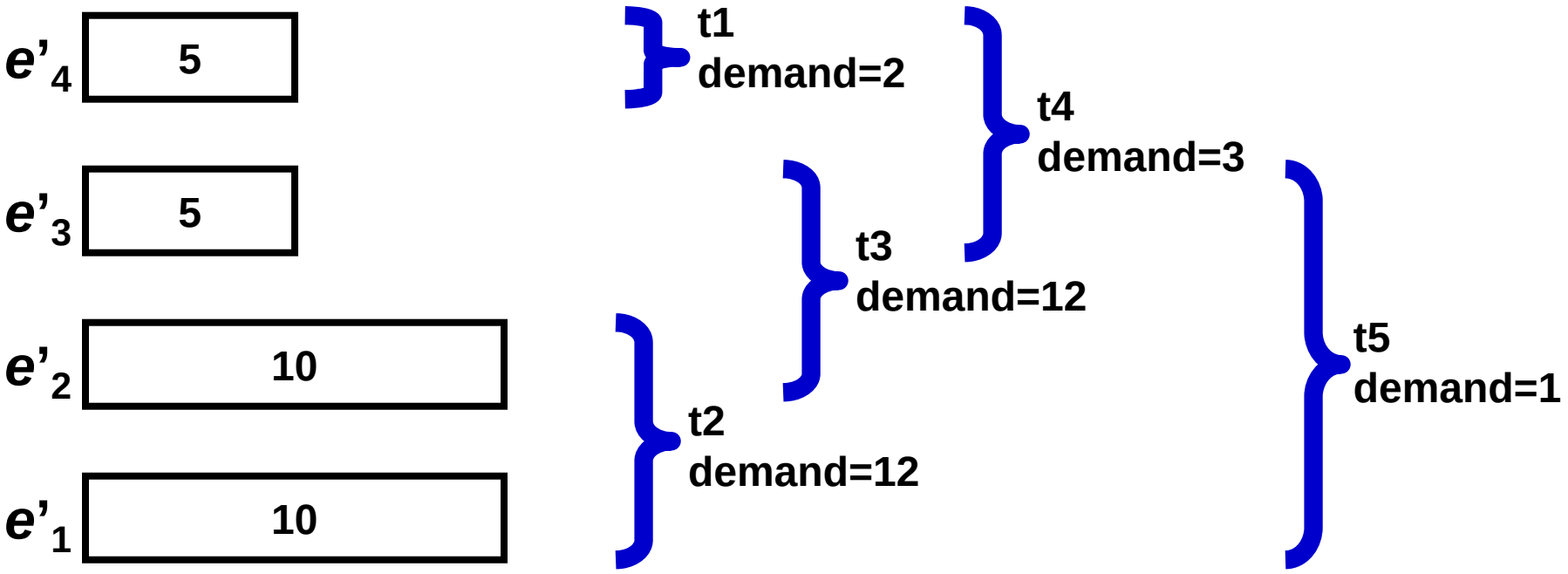
- Modifying the layer assignment method (COLA) of NTHU-Route 2.0 [TCAD'08]
  - Net ordering
    - LD nets appear before non-LD nets
  - Single-net layer assignment
    - Minimizing via count
    - Considering layer directives by adding penalty to the routing edges of LD nets which are not located in target layer ranges
- Keeping TOF identical to that of the 2D routing result

Our Router

- Enhancing GLADE to handle general layer directives during 2D global routing and layer assignment
  - Modifying the pseudo layer assignment method for calculating virtual demands
  - Adopting two-stage layer assignment without increase in TOF
    - Initial layer assignment for via count minimization
    - Iterative refinement for further minimizing LD violation and via count

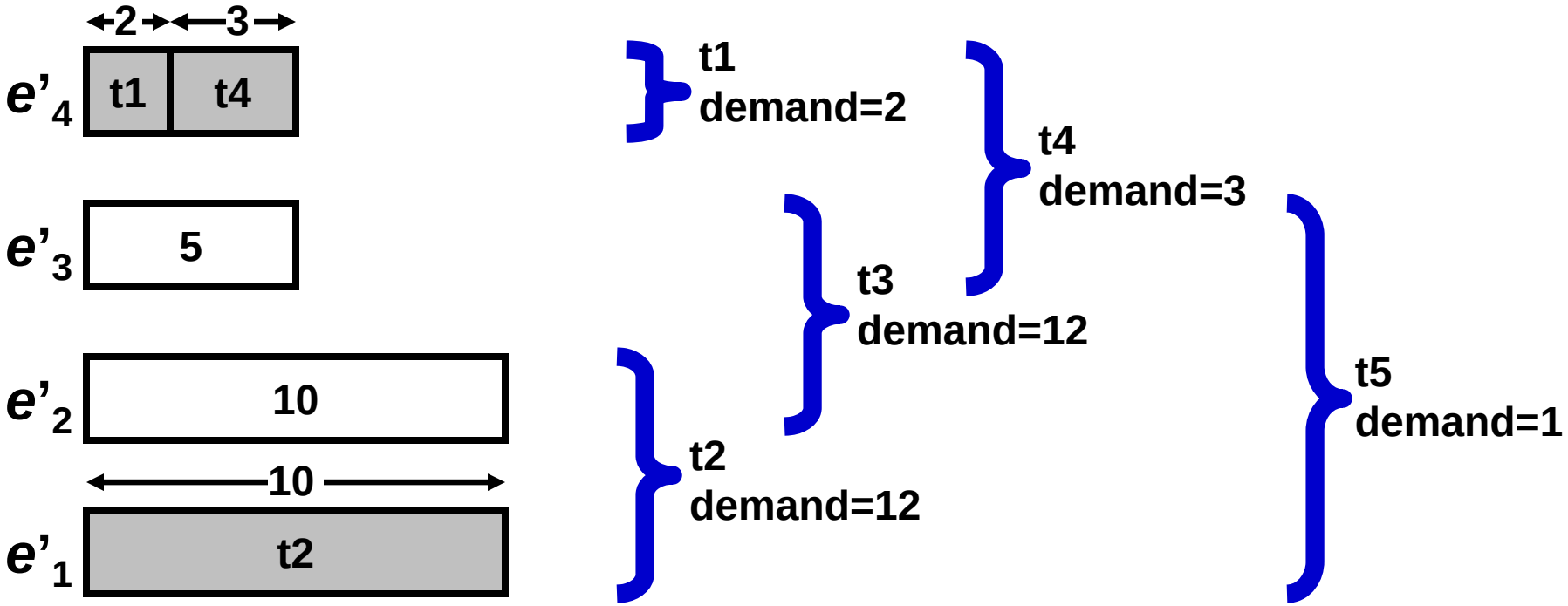
Calculation of VD (1/4)

- 3D edges e'_1, e'_2, e'_3 and e'_4 are projected to a 2D edge e
- We show how to calculate $vd_e(t5)$
- First, LD types are sorted in a non-decreasing order of the sizes of their layer ranges



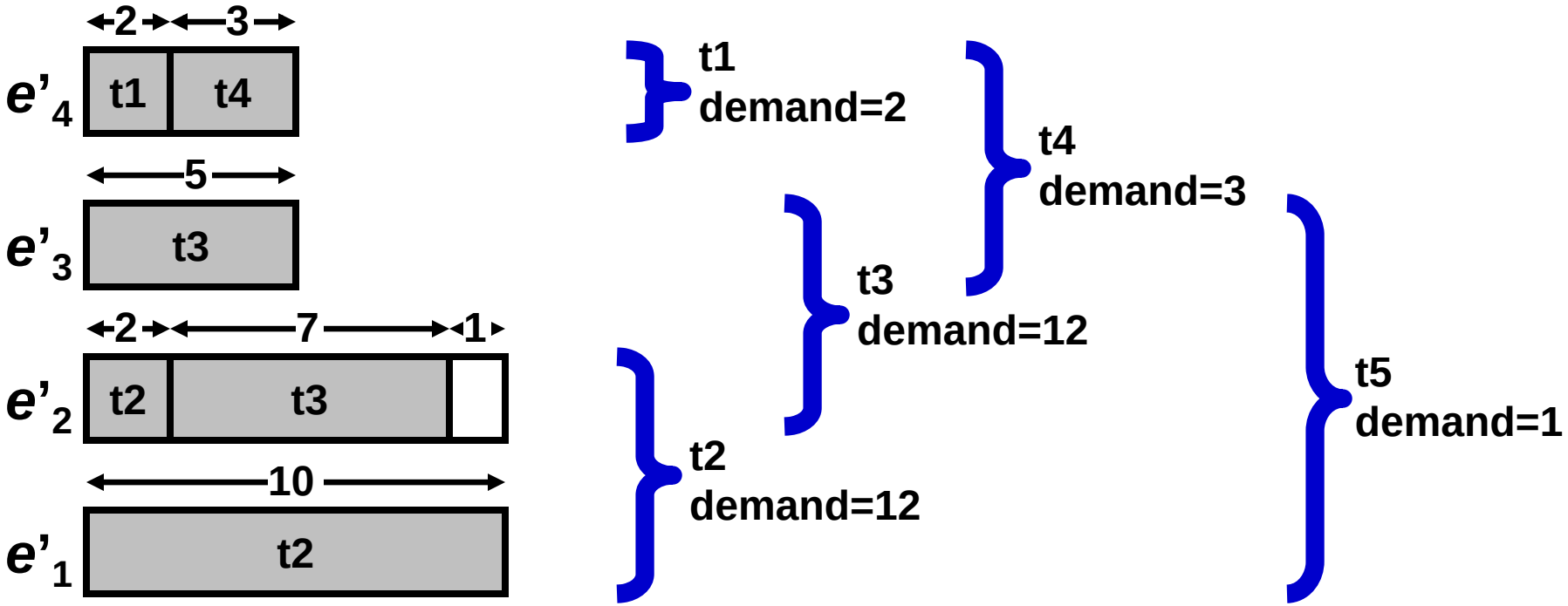
Calculation of VD (2/4)

- Step 1 (considering e'_4 and e'_1)
  - Assigning 2 nets of t1 and 3 nets of t4 to e'_4
  - Assigning 10 nets of t2 to e'_1



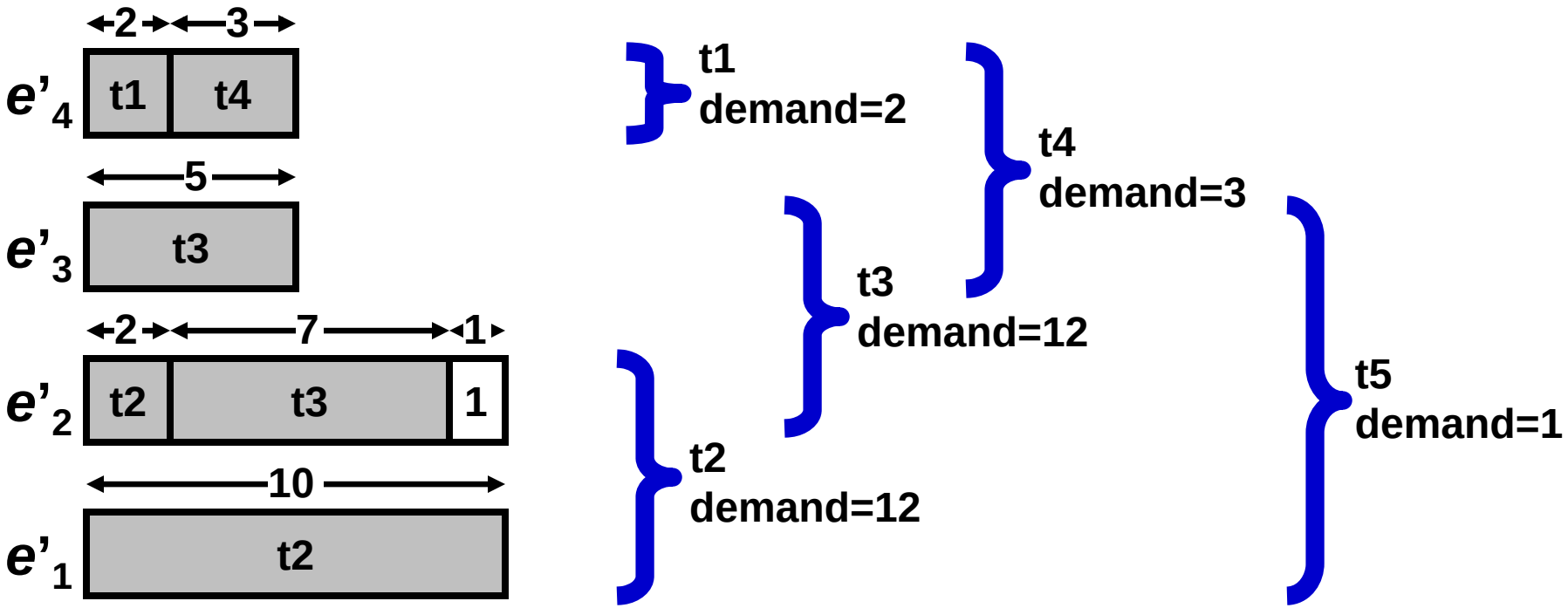
Calculation of VD (3/4)

- Step 2 (considering e'_3 and e'_2)
  - Assigning 5 nets of t3 to e'_3
  - Assigning 2 nets of t2 and 7 nets of t3 to e'_2



Calculation of VD (4/4)

- We get $vd_e(t5) = (5 + 2 + 7 + 10) + 1 = 25$

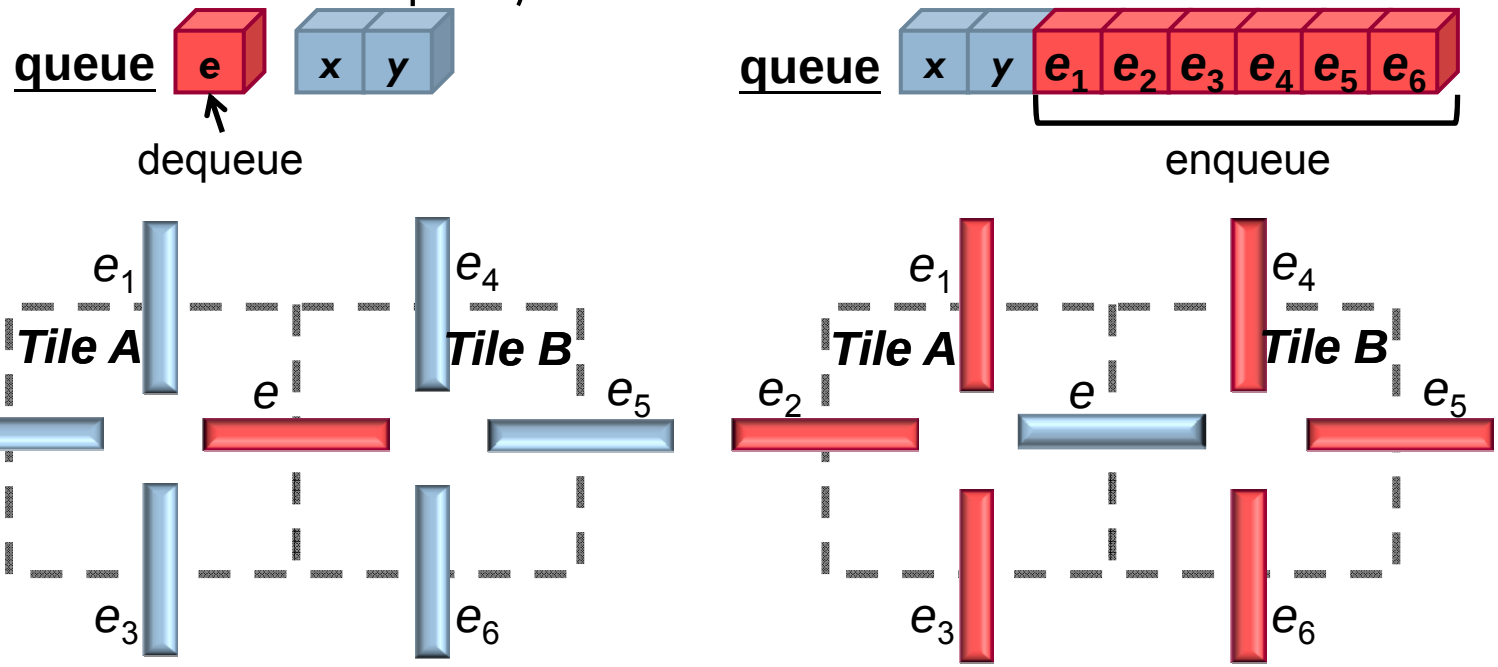


Two-Stage Layer Assignment: Initial Layer Assignment

- Adopting the layer assignment method COLA [TCAD'08] without considering layer directives
  - Targeting via count minimization
  - Keeping TOF identical to that of the 2D result

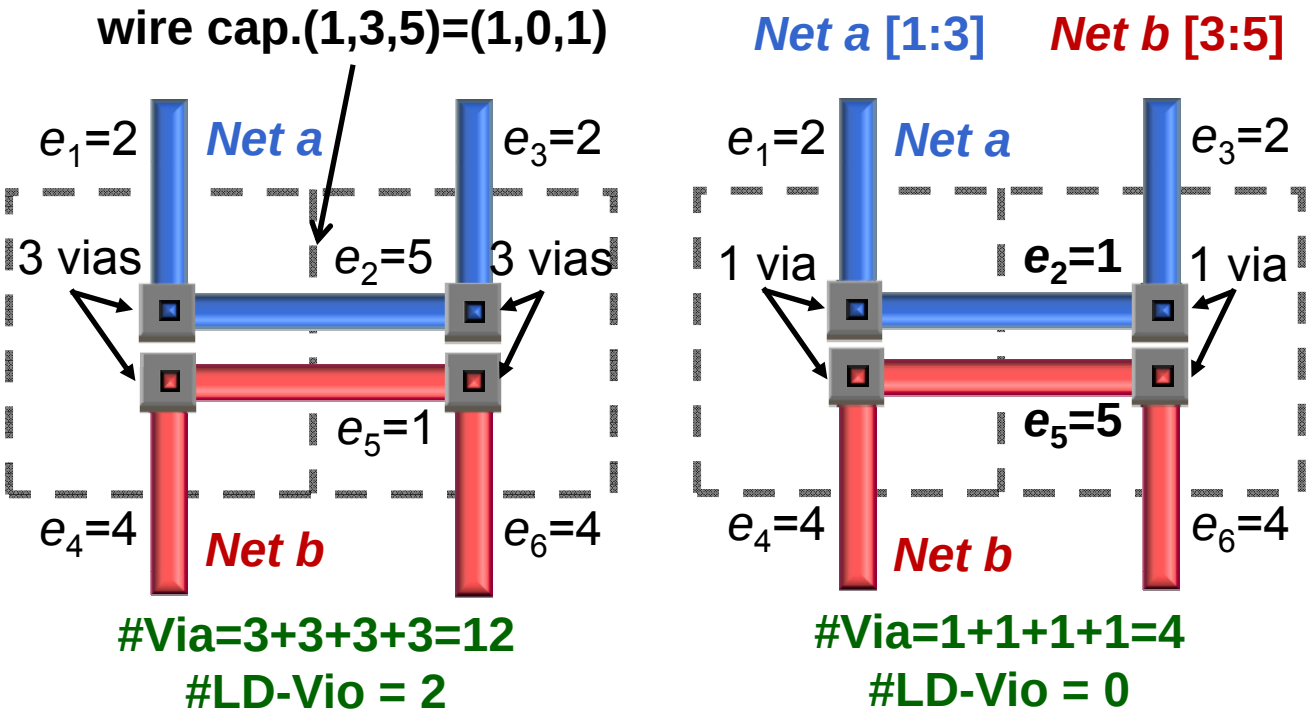
Two-Stage Layer Assignment: Refinement (1/5)

- Refining the solution for further minimization of LD violation and via count, but without TOF increase
  - Putting all 2D edges into a queue
  - Iteratively dequeuing an edge and applying a min-cost max-flow technique to re-assign its layer
    - If improved, accepting the result and enqueuing neighboring edges (if they are not in the queue)

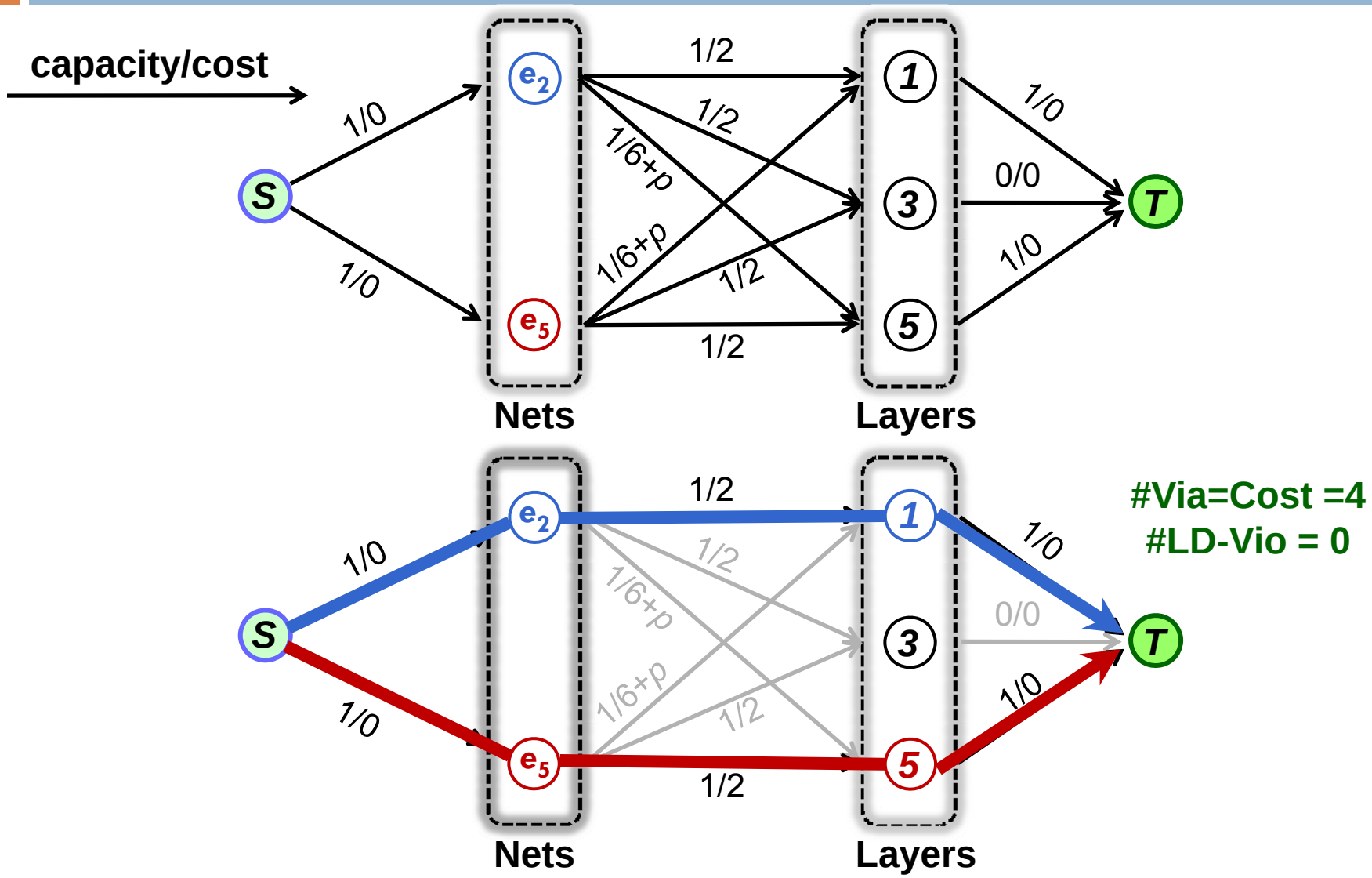


Two-Stage Layer Assignment: Refinement (2/5)

- 2D edge without overflow

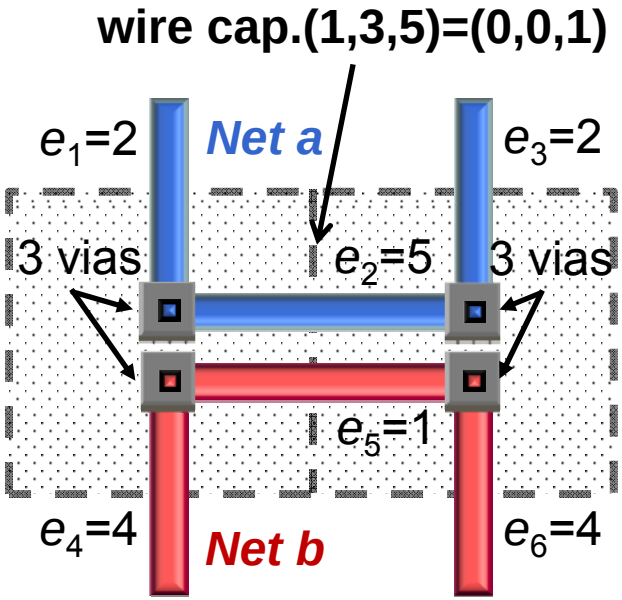


Two-Stage Layer Assignment: Refinement (3/5)

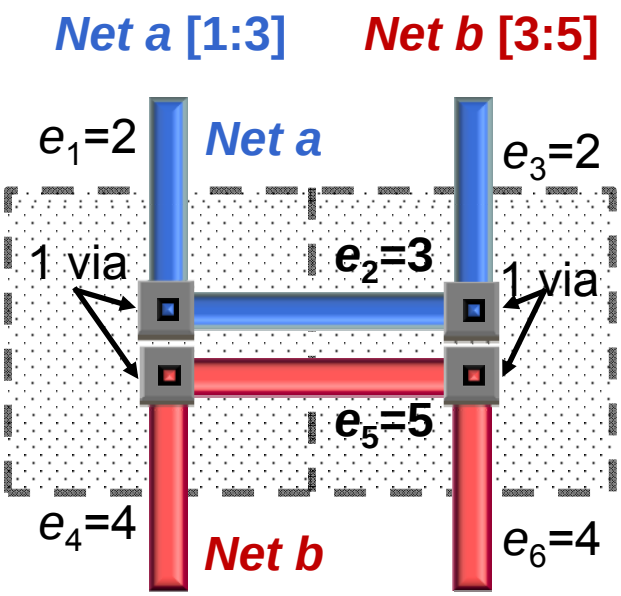


Two-Stage Layer Assignment: Refinement (4/5)

- 2D edge with overflow

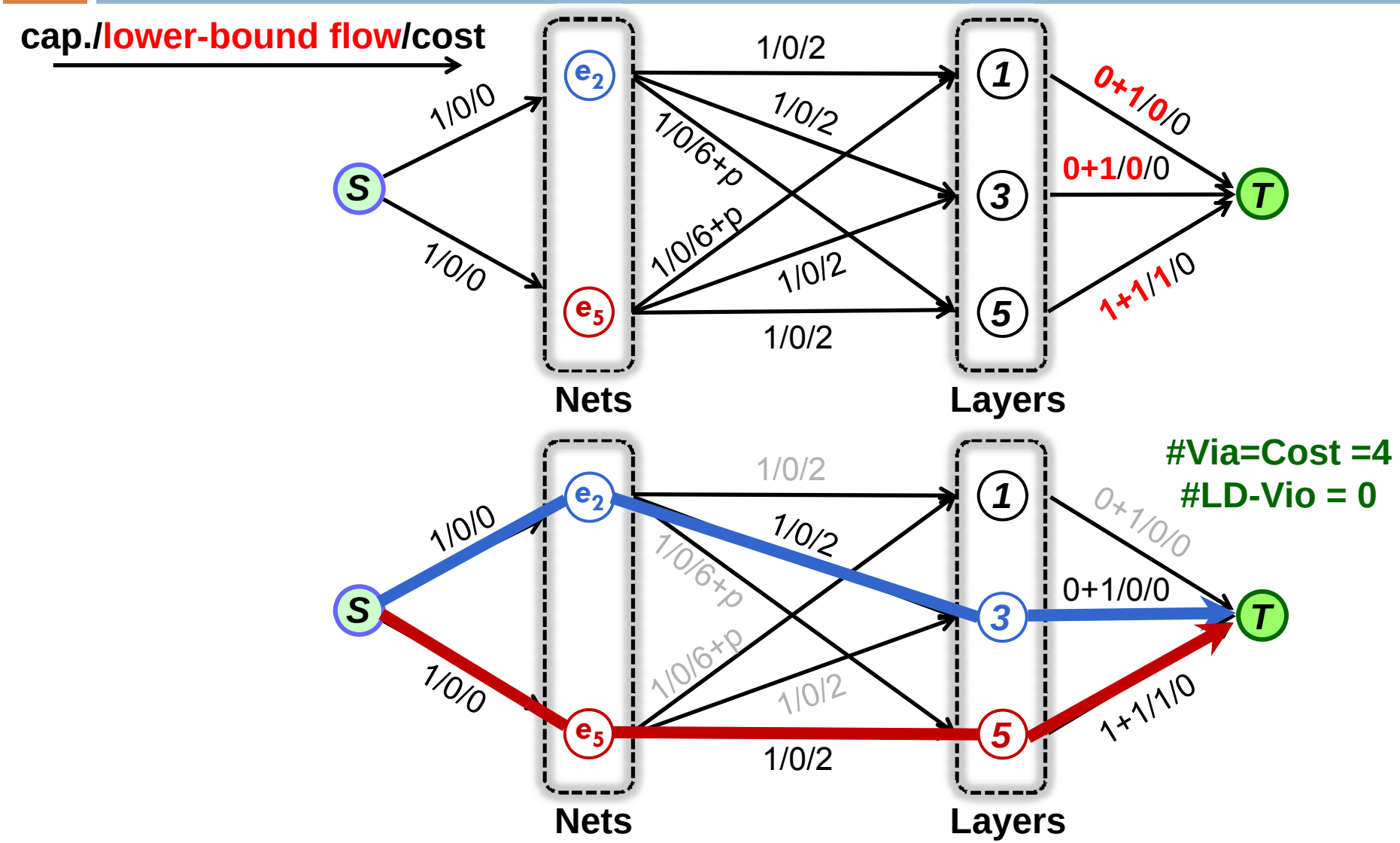


#Via=3+3+3+3=12
#LD-Vio = 2



#Via=1+1+1+1=4
#LD-Vio = 0

Two-Stage Layer Assignment: Refinement (5/5)



Experimental Results

- Our router was Implemented in C++
- All experiments were conducted on a Linux machine with Intel 2.2Ghz CPU and 8GB RAM
- Compared with two routers
  - GLADE
    - ICCAD 2009 benchmarks
  - NTHU-Route 2.0
    - Modified ICCAD 2009 benchmarks by randomly changing the layer ranges of LD nets

GLADE vs. Our Router

Bench-marks	GLADE					Our Router				
	TOF	LDOF	LD Vio	TWL	CPU	TOF	LDOF	LD Vio	TWL	CPU
adaptec1	0	0	0	45.4	7.0	0	0	59849/0	45.2/45.3	10.3
adaptec2	0	0	0	43.9	1.4	0	0	183623/0	43.2/43.8	4.2
adaptec3	0	0	0	115.2	7.2	0	0	210387/0	115.0/114.9	11.3
adaptec4	0	0	0	106.5	1.8	0	0	283214/0	105.9/106.5	3.9
adaptec5	0	0	0	130.1	15.2	0	0	66706/0	129.9/129.6	26.0
bigblue1	0	0	0	48.3	8.7	0	0	53858/0	48.5/48.5	17.1
bigblue2	0	0	0	69.6	7.0	0	0	7248/0	69.6/69.1	10.4
bigblue3	0	0	0	105.9	3.8	0	0	45669/0	105.7/105.5	10.4
bigblue4	188	0	0	178.9	121.0	188	0	71248/0	178.7/177.6	324.8
newblue1	2	0	0	35.6	4.8	2	0	6314/0	35.6/35.5	8.7
newblue2	0	0	0	59.7	0.8	0	0	49218/0	59.5/59.6	2.4
newblue4	140	0	0	108.1	40.1	140	0	45643/0	107.9/107.7	48.6
newblue5	0	0	0	190.7	12.6	0	0	9031/0	190.7/190.3	20.8
newblue6	0	0	0	139.8	11.5	0	0	26887/0	139.8/139.0	23.7
newblue7	78	0	0	281.7	119.9	78	0	113369/0	281.2/279/3	169.9
Comp.	--	--	1.000	1.000	1.000	--	--	--/1.000	0.998/0.996	1.904

NTHU-Route 2.0 vs. Our Router

Benchmarks	NTHU-Route 2.0				Our Router				
	TOF	LD Vio	TWL	CPU	TOF	LDOF	LD Vio	TWL	CPU
adaptec1	0	95066	45.1	5.3	0	0	46284/0	45.1/45.2	11.5
adaptec2	0	289132	43.1	1.3	0	0	145818/0	43.2/43.7	3.3
adaptec3	0	394924	114.9	5.6	0	0	205585/0	115.0/114.9	12.1
adaptec4	0	440412	105.9	1.7	0	0	222726/0	105.9/106.4	4.2
adaptec5	0	120402	129.8	15.7	0	0	59885/0	129.9/129.9	27.3
bigblue1	0	139562	47.8	7.0	0	0	58014/0	48.4/48.5	17.9
bigblue2	0	23070	69.3	6.0	0	0	11067/0	70.3/69.8	17.6
bigblue3	0	101772	105.7	3.7	0	0	54809/0	105.7/105.5	10.6
bigblue4	162	130542	178.7	75.8	236	0	66851/0	178.2/177.1	135.9
newblue1	0	13224	35.6	3.9	0	0	7513/0	35.6/35.5	8.1
newblue2	0	90746	59.4	0.9	0	0	44752/0	59.5/59.5	2.6
newblue4	138	73450	108.3	65.4	156	0	37856/0	107.7/107.5	51.0
newblue5	0	36910	190.7	12.8	0	0	19891/0	190.6/190.2	21.2
newblue6	0	36276	139.8	10.4	0	0	17986/0	139.8/139.0	25.4
newblue7	62	174794	279.8	57.8	82	0	84953/0	280.5/278.4	148.3
Comp.	--	1.000	1.000	1.000	--	--	0.502/0.000	1.001/0.998	1.818

Conclusion

- We have presented a global router that enhances a prior work, GLADE, to handle general layer directives.
- Encouraging experimental results have been provided to support our router.
- A possible future work is to improve our router for further reducing overflow values for benchmarks that are currently difficult to route.

THANK YOU

Q&A