Reinforcement Learning Guided Detailed Routing for Custom Circuits

Hao Chen[1], Kai-Chieh Hsu[2], Walker Turner[3], Po-Hsuan Wei[3], Keren Zhu[1],

David Z. Pan[1], and Haoxing Ren[3]

[1]ECE Department, The University of Texas at Austin

[2]ECE Department, Princeton University

[3]NVIDIA Corporation

This work is supported in part by NVIDIA, the NSF under Grant No. 1704758, and the DARPA ERI IDEA program

Outline

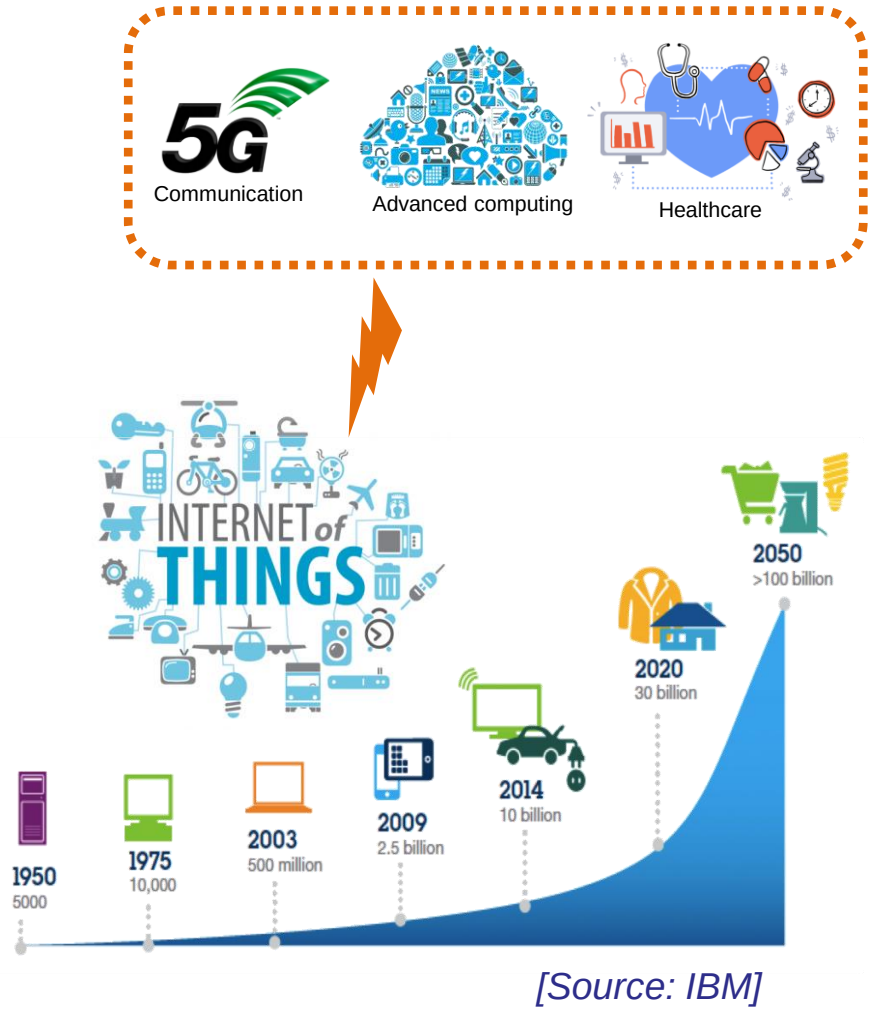
- Background
- RL Guided Custom Detailed Routing Framework
- Experimental Results
- Conclusion and Future Work

Outline

- **Background**
- RL Guided Custom Detailed Routing Framework
- Experimental Results
- Conclusion and Future Work

Custom Circuits are Everywhere

- Analog/mixed-signal (AMS) circuits
  - Internet of Things (IoT), autonomous vehicles
  - **Every sensor-related application**
- Sensitive digital circuits
  - High speed/performance SerDes
- Interconnect circuits
  - Electro-optical links

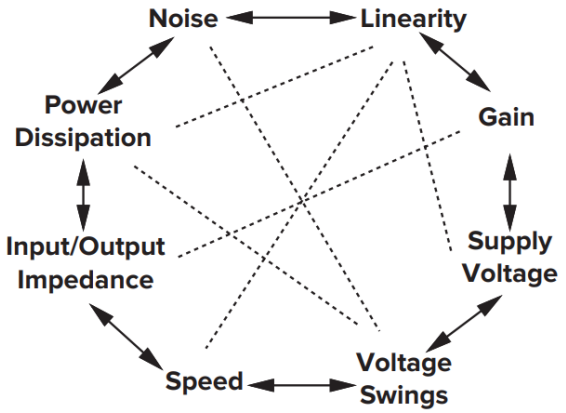


[Source: IBM]

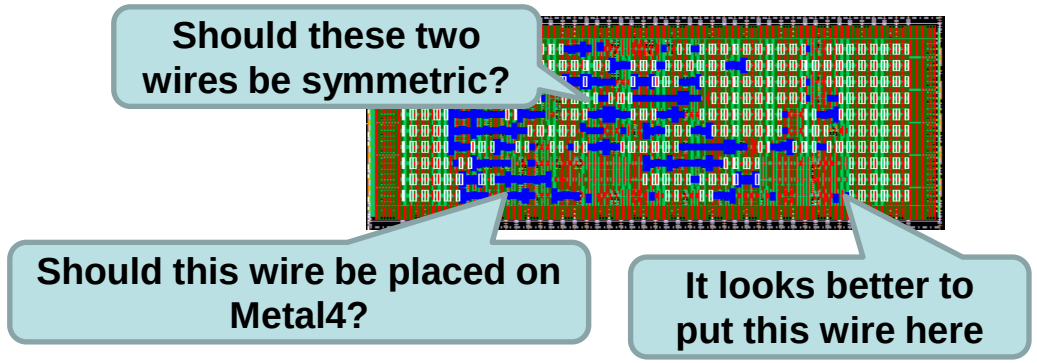
Custom layout is a bottleneck in chip design!!

Challenges of Custom Routing

- **Sensitive nature**
  - Suspectable to parasitics
- **Complicated design rules**
- **Layout constraints**
  - Geometrical/Electrical constraints
- **No common evaluation metrics**
- **Design conventions and aesthetic engineering**



[Source: Razavi, Design of Analog IC]



Aesthetic is often a surrogate for correctness

[Rutenbar, 2016]

Existing Custom Routing Algorithms

- **Template-based routing** *[Crossley+,ICCAD'13],…*
- **Simulation-based routing** *[Choudhury+, DAC'90],…*
- **Constrained routing**
  - Symmetry constraints: [*Xiao+,ICCAD'10*], [*Chen+, ICCAD'20*], …
  - Topology-matching constraints: [*Yan+, ICCAD'08*], [*Ou+, TCAD'14*], …
  - Exact-matching constraints: [*Ozdal+, ICCAD'12*], …

Designed for specific use cases

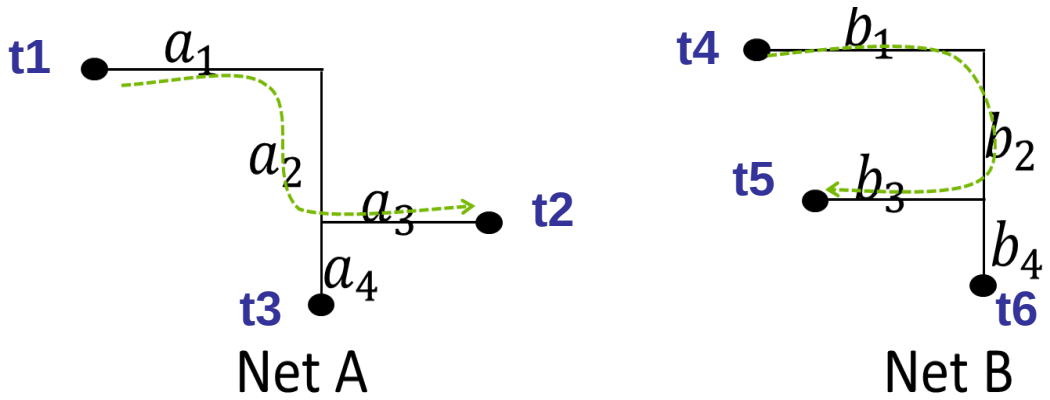
- Some are applied in planar technology nodes or PCB
- Hard to cover all kinds of designs
- Cannot be easily modified and extended for future needs

We want a constraint with good flexibility and descriptiveness

Path-Matching Constraints

- A path is a pin-to-pin connection within a net
- A path-matching constraint consists of several paths to be matched
  - Paths can be in the same net or different nets
  - Match the resistance of paths in the same constraint
- Can describe widely used geometrical constraints (e.g., symmetry, exact-matching)

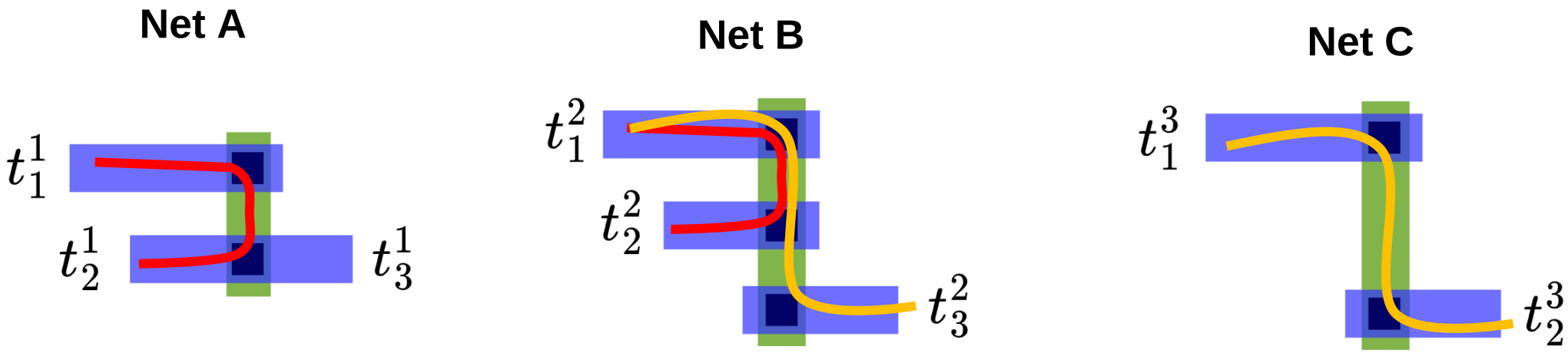
Constraint: Match (t1 → t2), (t4 → t5)
2 paths, 1 matching group



Make the two path resistances as close as possible

Solving Path-Matching Constraints

- **Hard problem**
  - No trivial heuristic algorithms
  - Different path-matching constraints can be dependent



Matching the red paths could affect the matching of yellow paths

RL!!

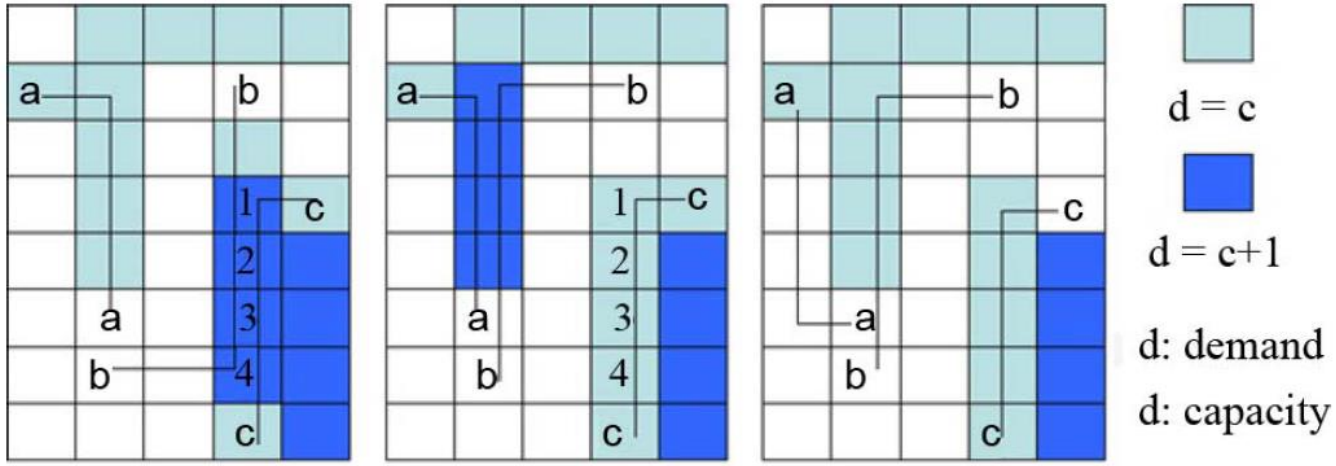
Outline

- Background
- **RL Guided Custom Detailed Routing Framework**
- Experimental Results
- Conclusion and Future Work

Rip-Up and Re-Route

- Widely adopted in modern routers to handle congestion
  - Basic rip-up and re-route
    - Remove all violated nets **(inefficient and hard to converge)**
  - Negotiation-based rip-up and re-route (NRR)
    - Maintain history costs on routing grid edges

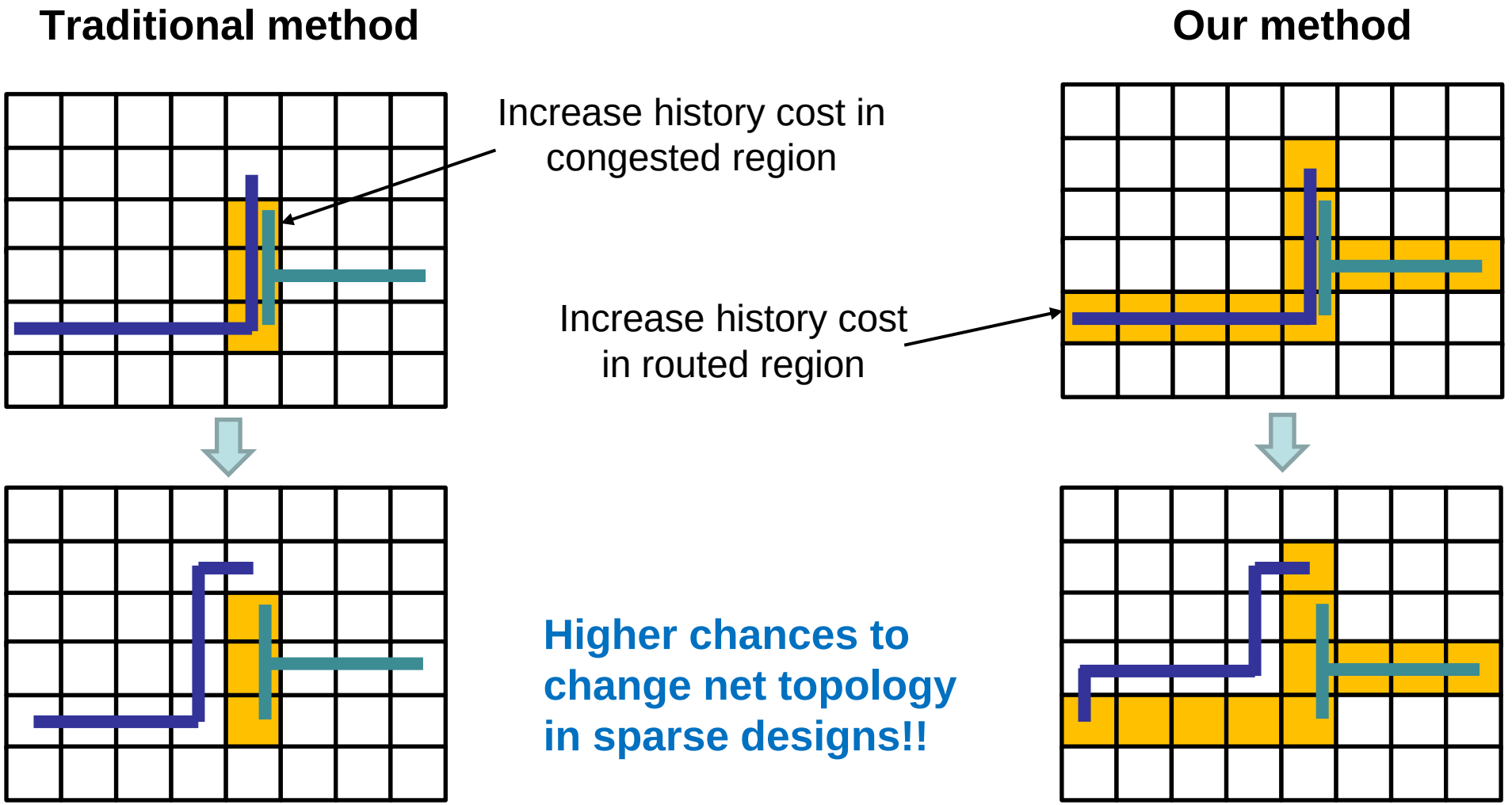
Example: L-shape routing with history



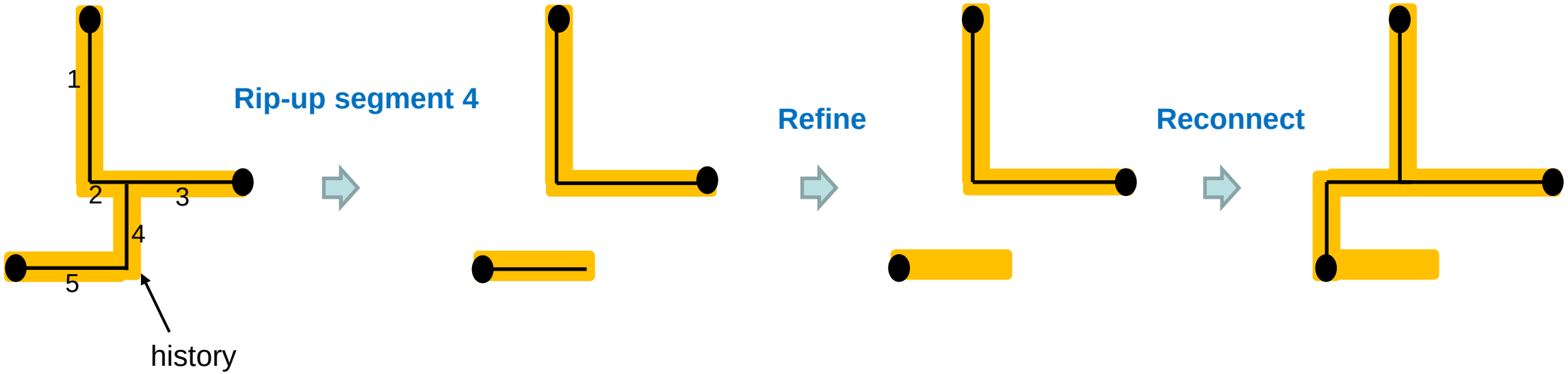
[Source: Liu+, TCAD'12, NCTU-GR]

Routing Topology Adjustment

- Use a revised history cost method to encourage new routing topologies

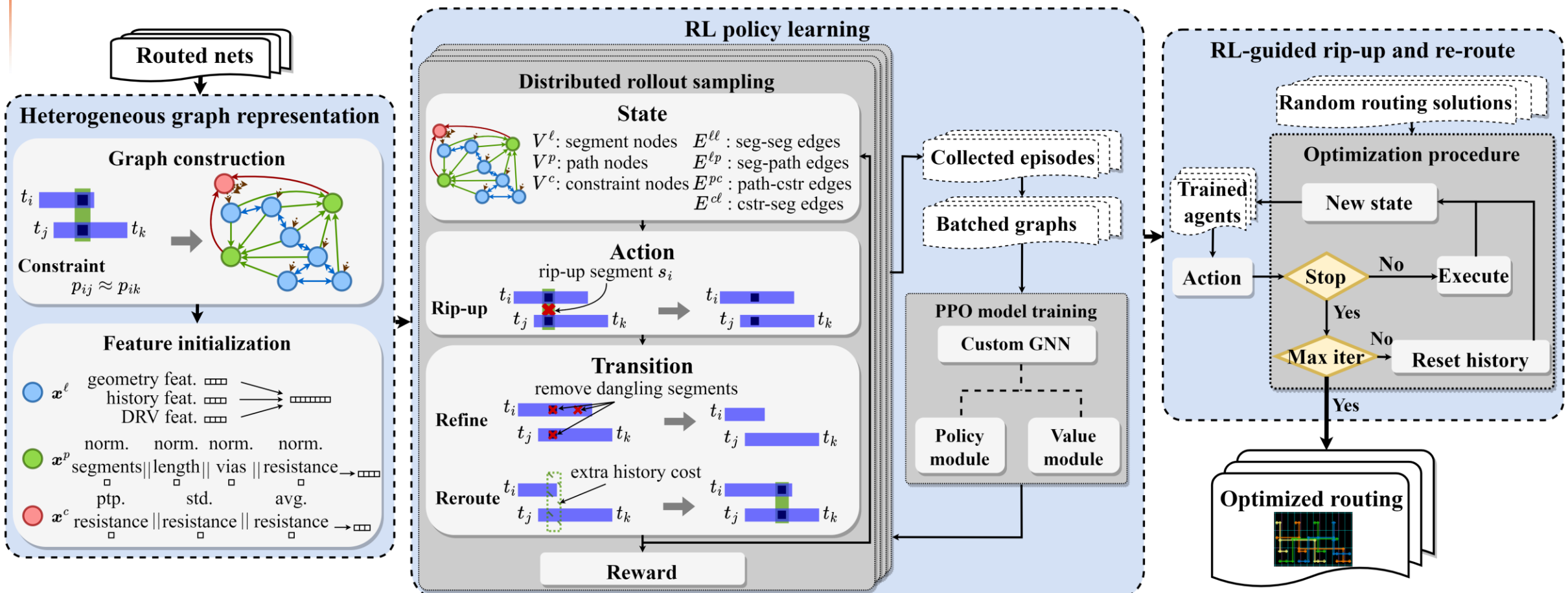


Routing Topology Adjustment

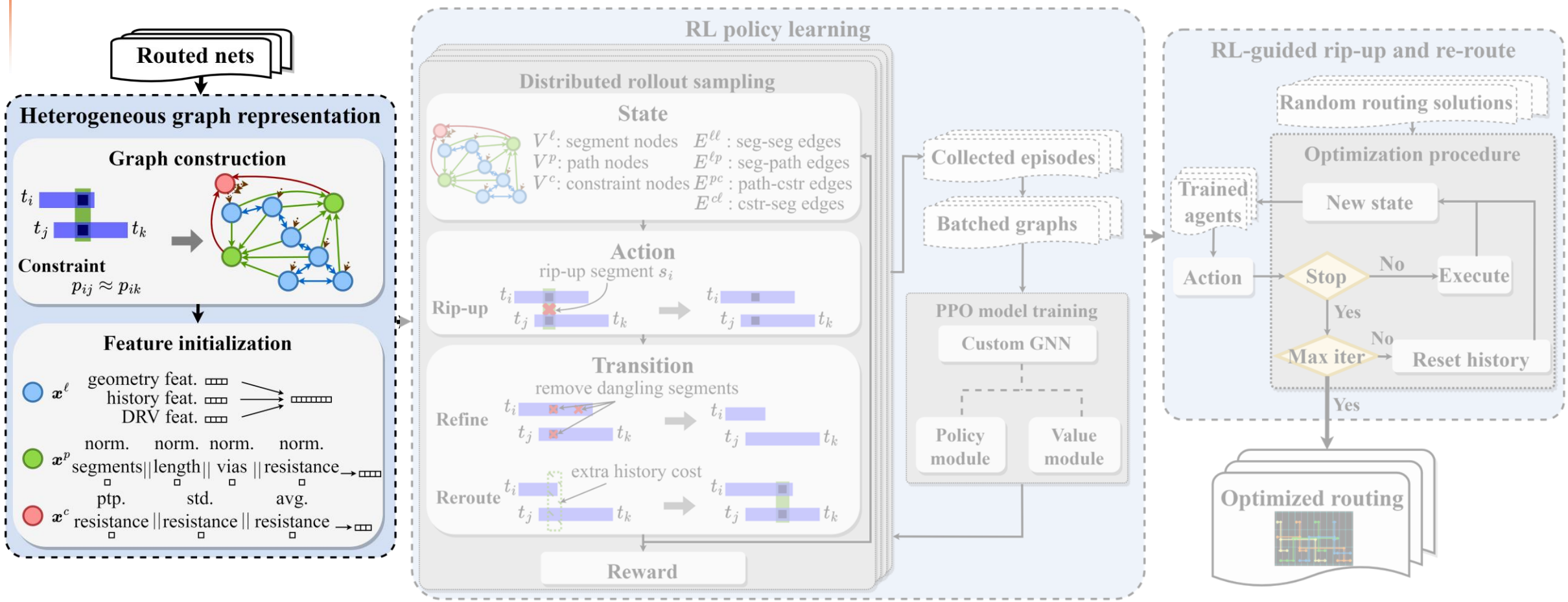


Repeat the process to reach the desired routing topology

RL Guided Custom Detailed Routing Framework



RL Guided Custom Detailed Routing Framework



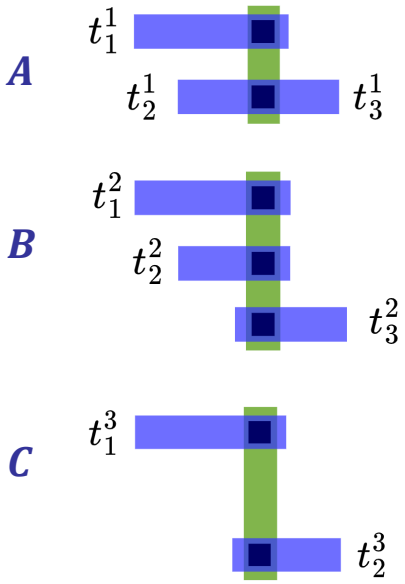
Heterogeneous Graph Representation

- Routing solutions are transformed into heterogeneous graphs
  - Nodes: segment nodes, path nodes, constraint nodes
  - Edges: seg-seg edges, seg-path edges, path-cstr edges, cstr-seg edges

Nets A, B, C **Constraints**

- c_1: match paths $(t_1^1 \rightarrow t_2^1)$, $(t_1^1 \rightarrow t_3^1)$, $(t_1^2 \rightarrow t_2^2)$
- c_2: match paths $(t_1^2 \rightarrow t_3^2)$, $(t_1^3 \rightarrow t_2^3)$

2 constraints
5 paths



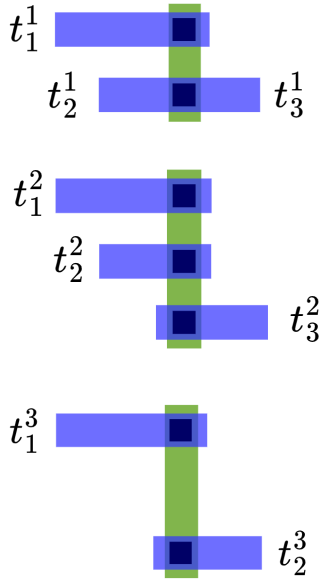
Heterogeneous Graph Representation

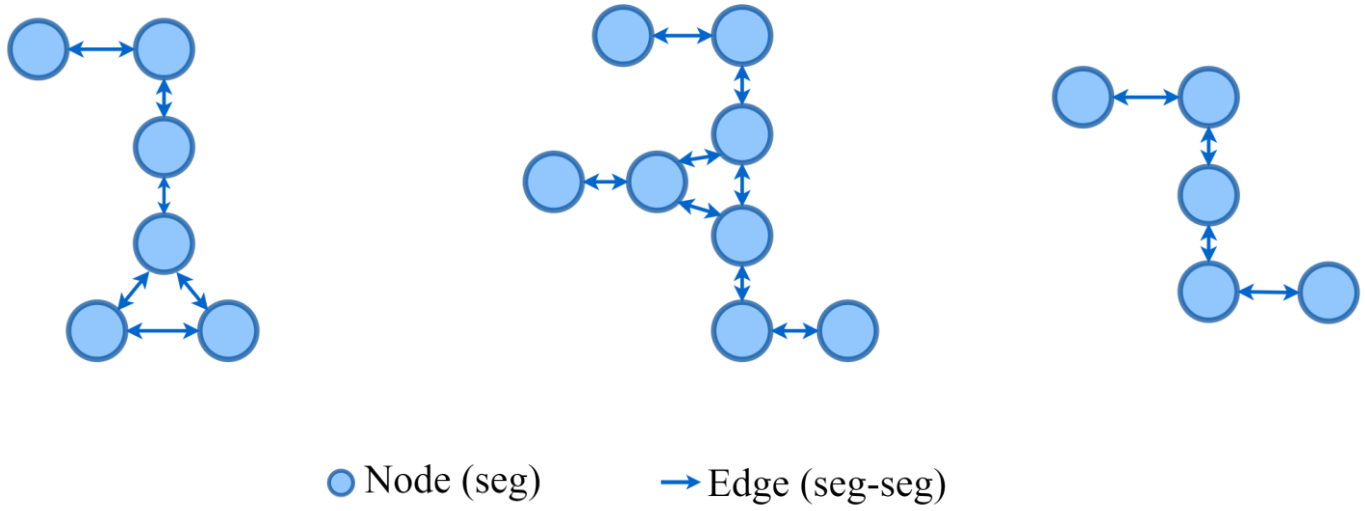
- Routing solutions are transformed into heterogeneous graphs
  - Nodes: segment nodes, path nodes, constraint nodes
  - Edges: seg-seg edges, seg-path edges, path-cstr edges, cstr-seg edges

Nets A, B, C **Constraints**

- c_1: match paths $(t_1^1 \rightarrow t_2^1)$, $(t_1^1 \rightarrow t_3^1)$, $(t_1^2 \rightarrow t_2^2)$
- c_2: match paths $(t_1^2 \rightarrow t_3^2)$, $(t_1^3 \rightarrow t_2^3)$

2 constraints
5 paths





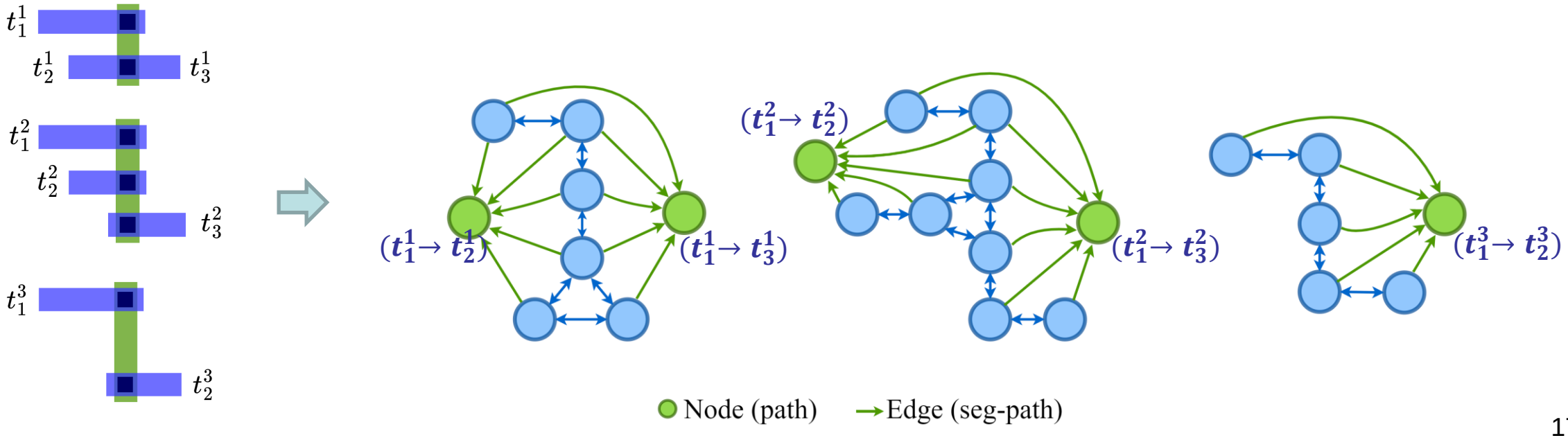
Heterogeneous Graph Representation

- Routing solutions are transformed into heterogeneous graphs
  - Nodes: segment nodes, path nodes, constraint nodes
  - Edges: seg-seg edges, seg-path edges, path-cstr edges, cstr-seg edges

Nets A, B, C **Constraints**

- c_1: match paths $(t_1^1 \rightarrow t_2^1)$, $(t_1^1 \rightarrow t_3^1)$, $(t_1^2 \rightarrow t_2^2)$
- c_2: match paths $(t_1^2 \rightarrow t_3^2)$, $(t_1^3 \rightarrow t_2^3)$

2 constraints
5 paths



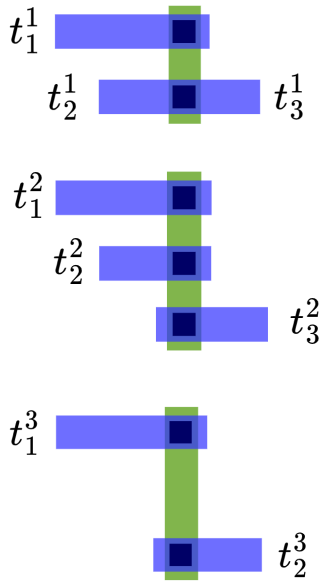
Heterogeneous Graph Representation

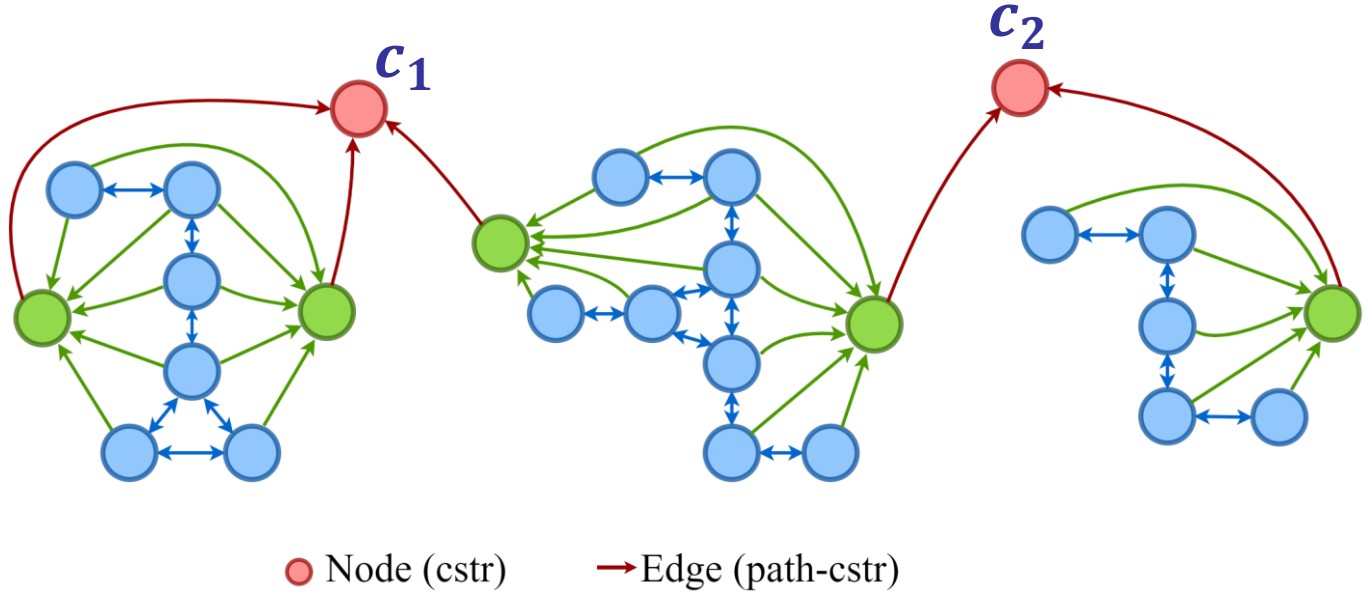
- Routing solutions are transformed into heterogeneous graphs
  - Nodes: segment nodes, path nodes, constraint nodes
  - Edges: seg-seg edges, seg-path edges, path-cstr edges, cstr-seg edges

Nets A, B, C **Constraints**

- c_1: match paths $(t_1^1 \rightarrow t_2^1)$, $(t_1^1 \rightarrow t_3^1)$, $(t_1^2 \rightarrow t_2^2)$
- c_2: match paths $(t_1^2 \rightarrow t_3^2)$, $(t_1^3 \rightarrow t_2^3)$

2 constraints
5 paths





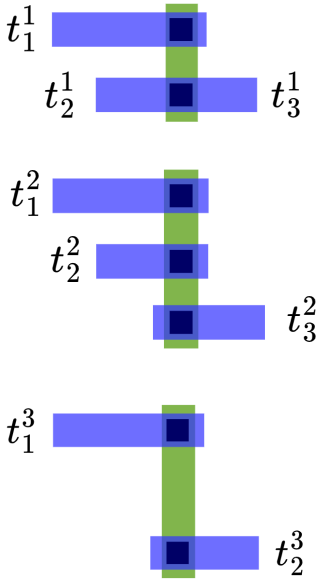
Heterogeneous Graph Representation

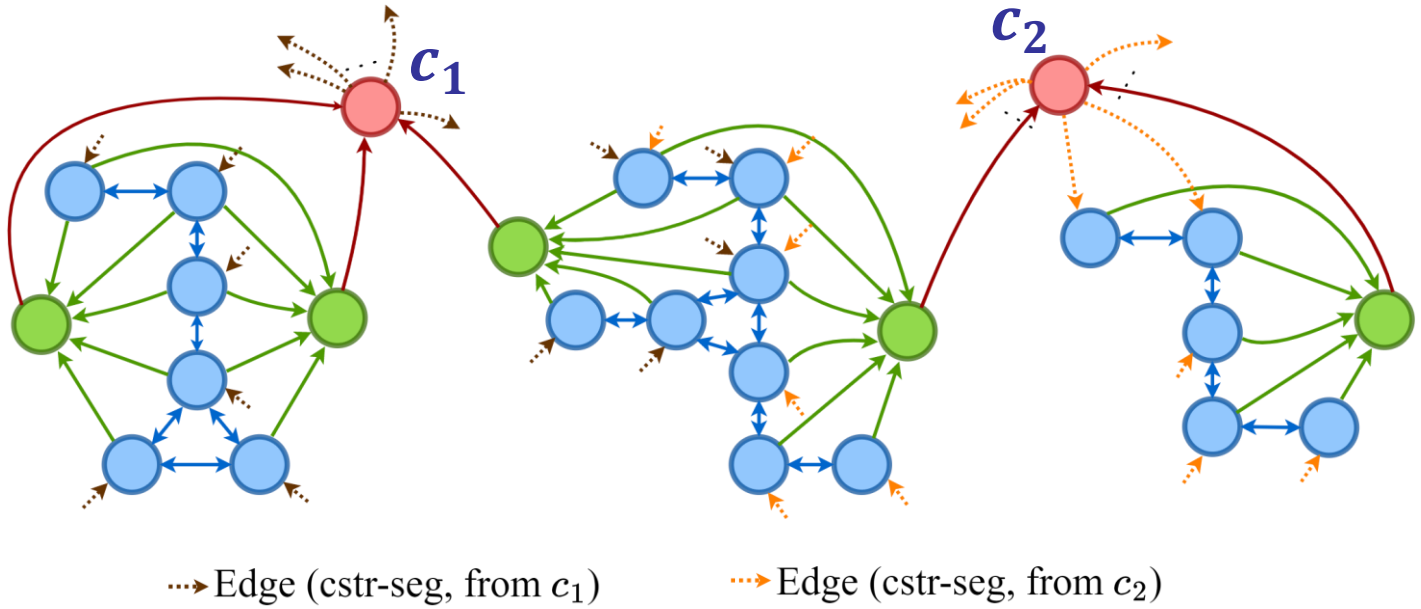
- Routing solutions are transformed into heterogeneous graphs
  - Nodes: segment nodes, path nodes, constraint nodes
  - Edges: seg-seg edges, seg-path edges, path-cstr edges, cstr-seg edges

Nets A, B, C **Constraints**

- c_1: match paths $(t_1^1 \rightarrow t_2^1)$, $(t_1^1 \rightarrow t_3^1)$, $(t_1^2 \rightarrow t_2^2)$
- c_2: match paths $(t_1^2 \rightarrow t_3^2)$, $(t_1^3 \rightarrow t_2^3)$

2 constraints
5 paths

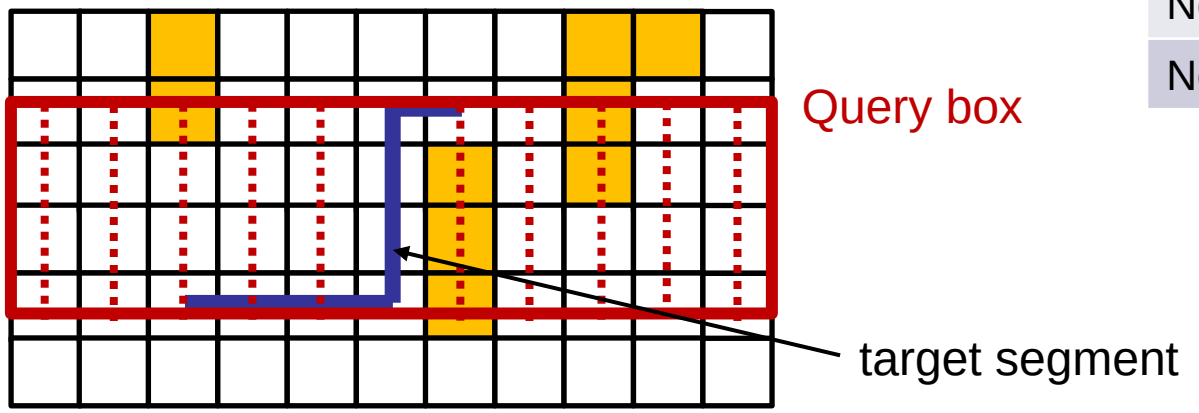




Graph Node Features

Segment nodes

Feature Description	Dimension
Norm. bottom-left coordinate (x, y)	2
Norm. top-right coordinate (x, y)	2
Norm. length	1
Norm. layer index (lower layer for vias)	1
Norm. design rule violation count	1
Norm. neighboring history costs	11
Boolean indicator for via	1
Boolean indicator for terminal segment	1



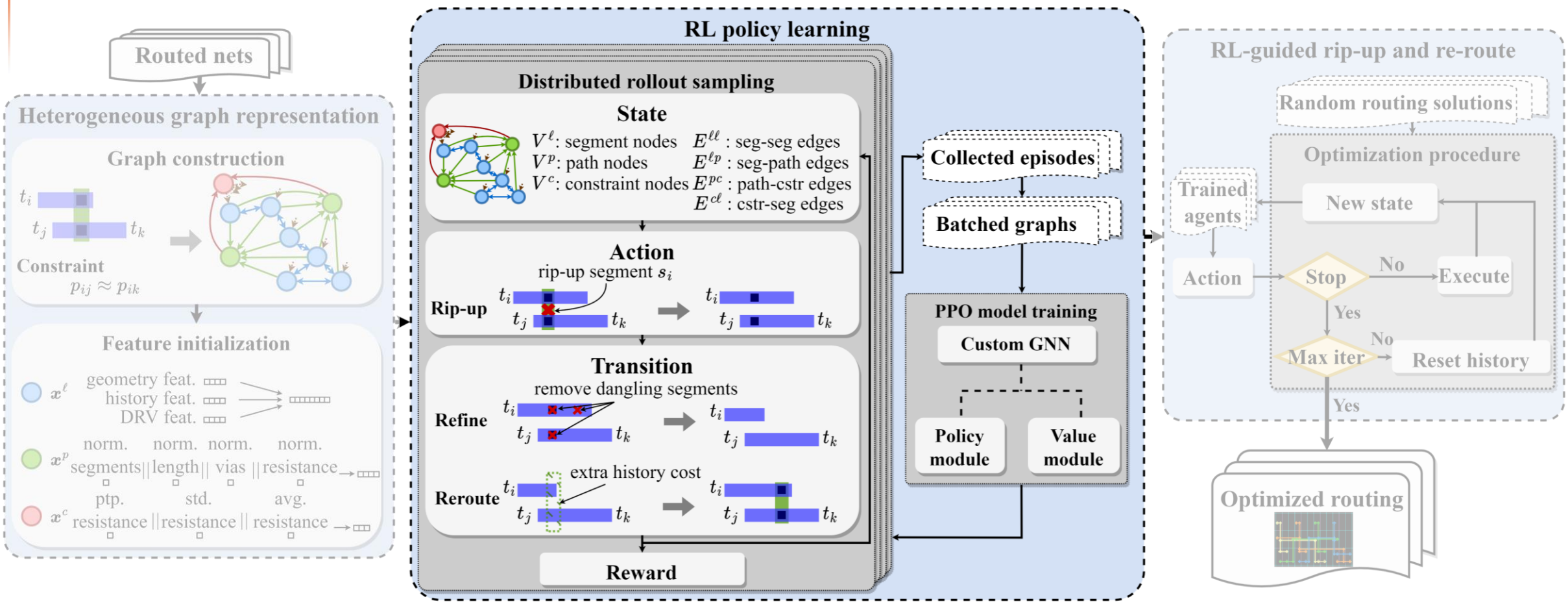
Path nodes

Feature Description	Dimension
Norm. segment count	1
Norm. length	1
Norm. via count	1
Norm. path resistance	1

Constraint nodes

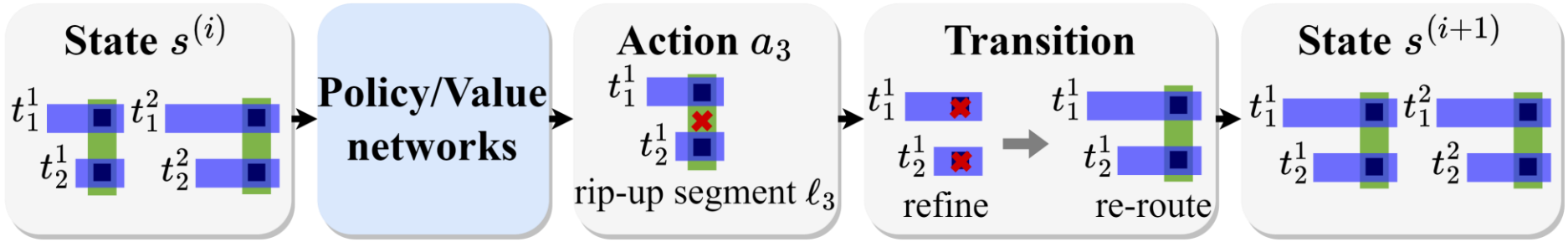
Feature Description	Dimension
Norm. maximum difference of path resistances	1
Norm. average of path resistances	1
Norm. std. of path resistances	1

RL Guided Custom Detailed Routing Framework

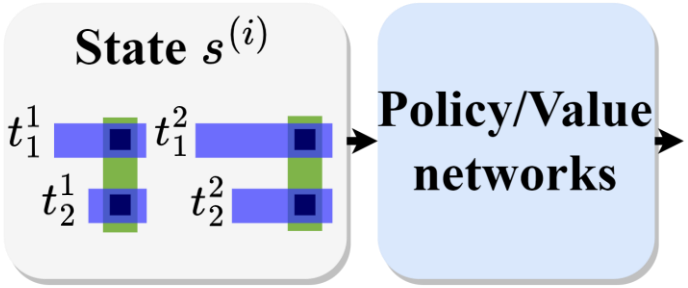


Actions

- Choose a segment to remove



- Or do nothing (stop the Markov process)

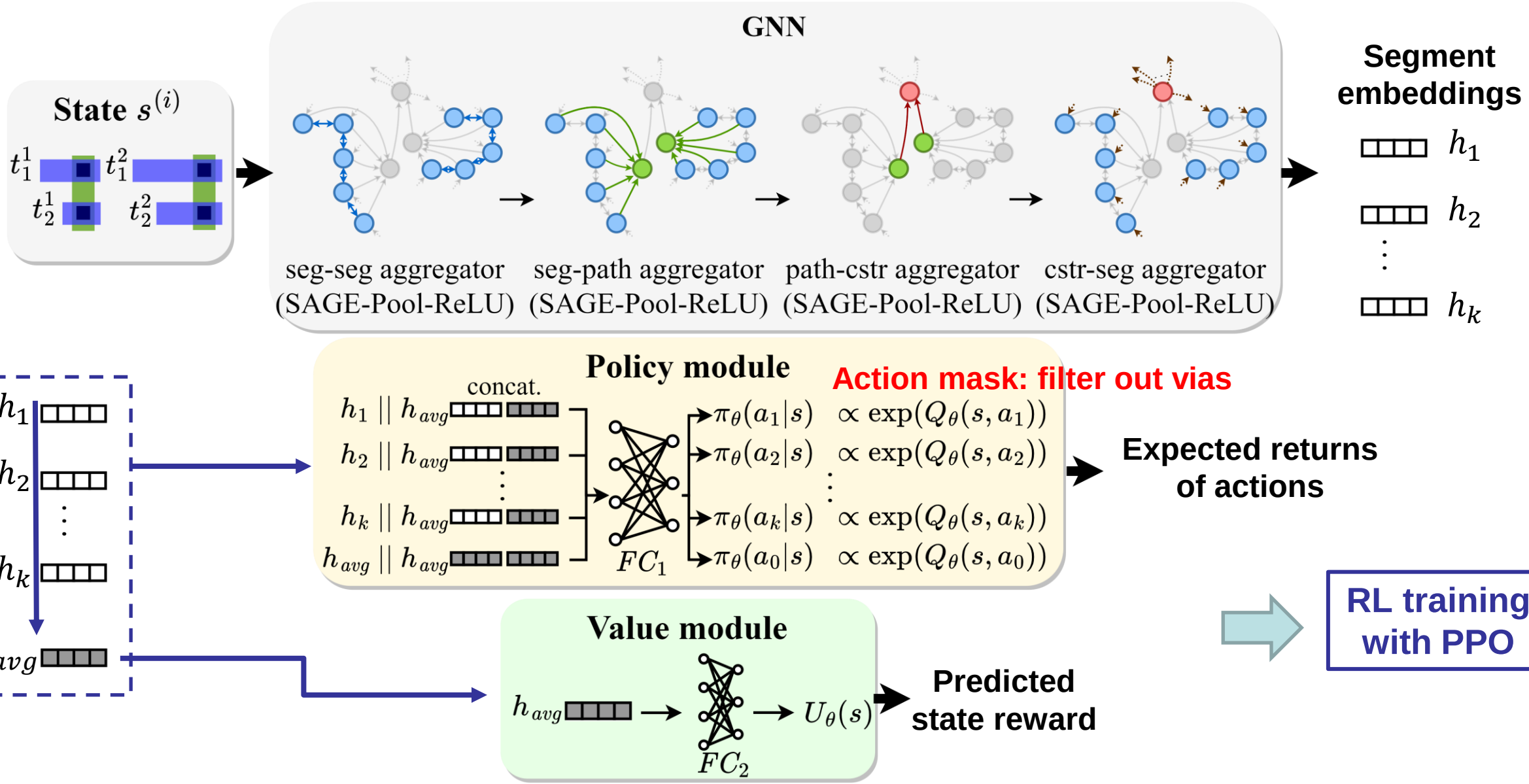


DONE!!

Sequence length matters!!

- **Too short: optimized solutions not yet reached**
- **Too long: accumulated history costs block the agent from getting good solutions**

Policy/Value Network Architecture



Reward Function

- Objectives
  - Total wirelength
  - Total via count
  - Total design rule violations
  - Avg. (peak-to-peak path resistance difference) of all path-matching constraints

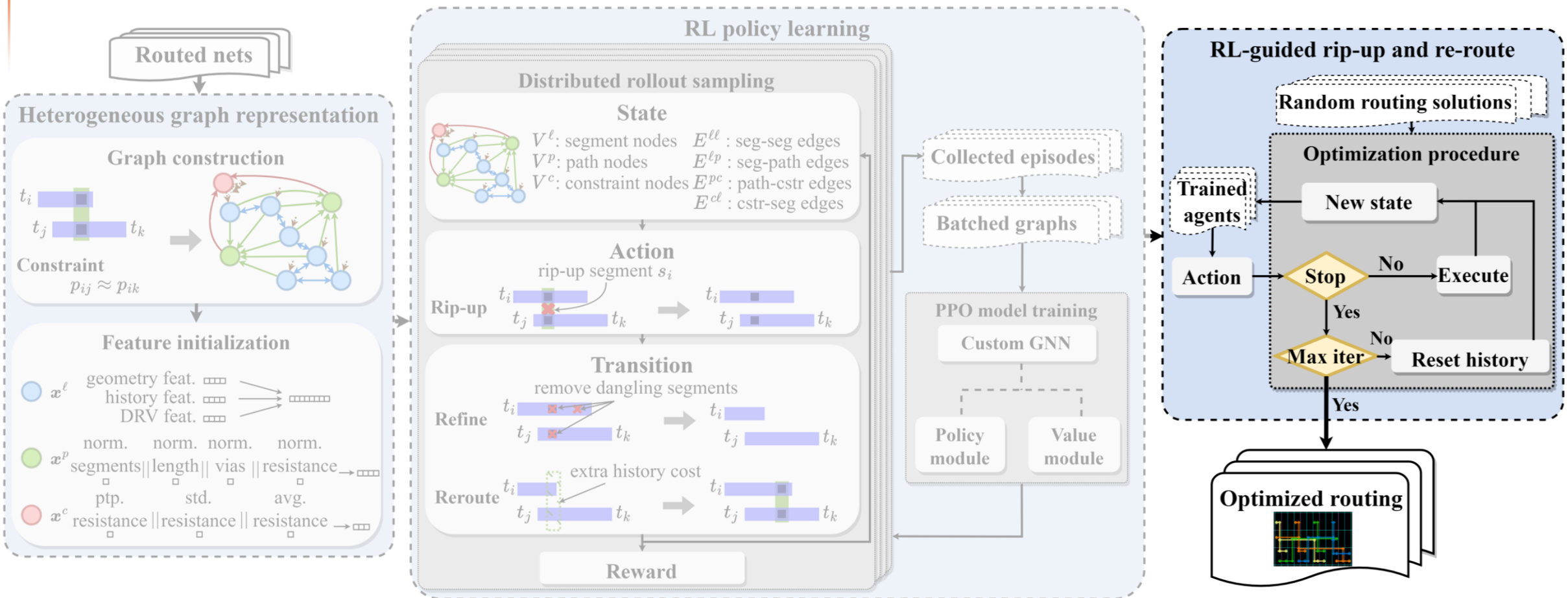
$$w_{wl}(\text{previous WL} - \text{current WL})$$ **Wirelength improvement of the action**

$$+w_{via}(\text{previous VIA} - \text{current VIA})$$ **Via improvement of the action**

$$+w_{drv}(\text{previous DRV} - \text{current DRV})$$ **DRV improvement of the action**

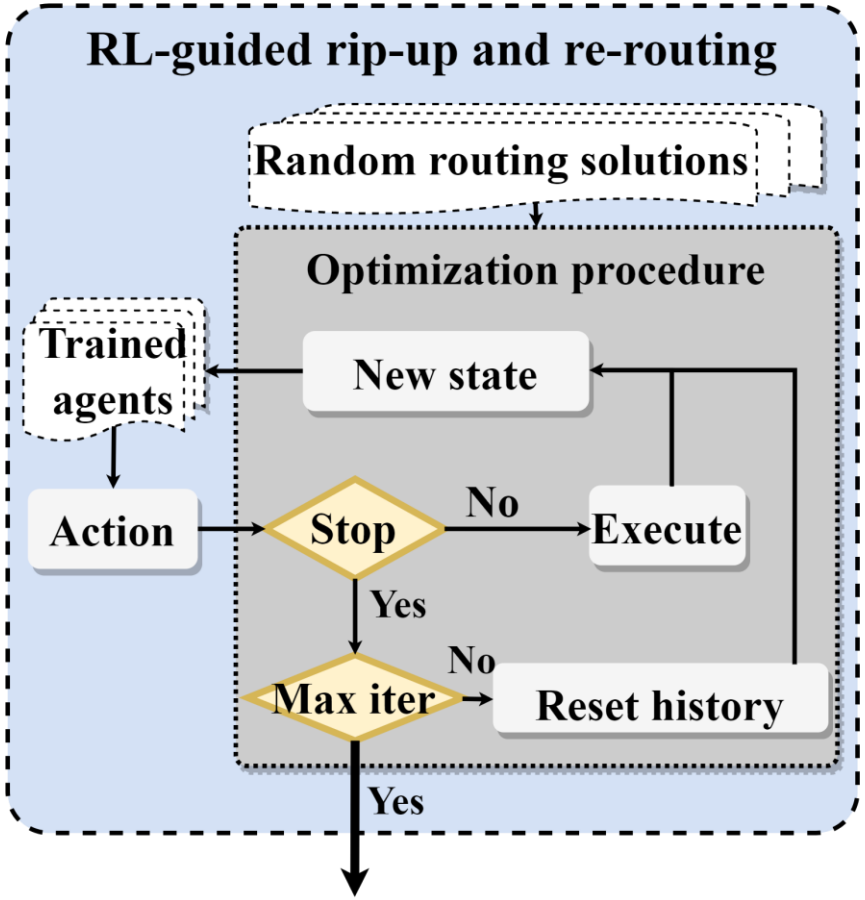
$$+w_{ptp}(\text{previous PTP} - \text{current PTP})$$ **Matching improvement of the action**

RL Guided Custom Detailed Routing Framework



RL-Based Rip-Up and Re-Route

- Integrated with trained RL policies
  - Start with randomly routed solutions
  - Optimize routing patterns for each match group sequentially
    - RL agent decides which segment to remove, or stops the process
  - Reset the history costs and repeat the procedure to further optimize solutions



DONE!!

Outline

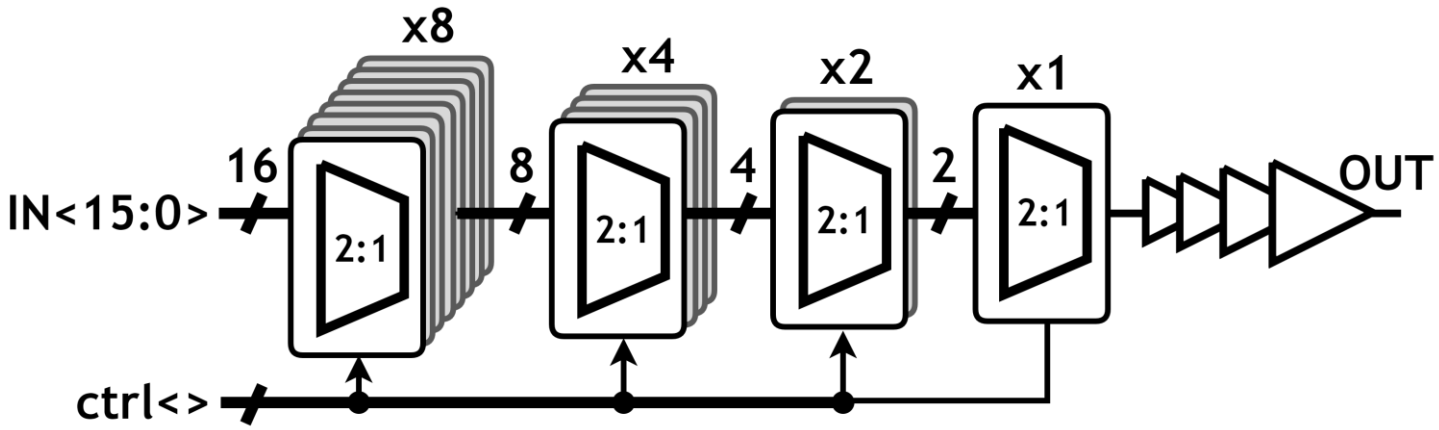
- Background
- RL Guided Custom Detailed Routing Framework
- **Experimental Results**
- Conclusion and Future Work

Experiment Settings

- Comparison of layouts
  - Manual
  - AutoCRAFT [Chen+, ISPD'22]
  - RL (this work)
- Same placement for each test circuit
- Additional net symmetry constraints are specified for AutoCRAFT's custom router

16:1 Multiplexing Buffer (BUF)

- Designed to drive a large capacitive load with the ability to select between 16 input signals (IN<15:0>) using 4-bit control (ctrl<>)
  - **42 cells, 66 nets, 228 pins**
  - **6 path-matching constraints, 30 paths**
  - **Manual placement**



Post-Layout Performance of BUF

Metric	AutoCRAFT	RL
WL (μm)	96.6	102.0
VIA	265	272
DRV	0	0
PTP (Ω)	64.6	**22.3**
Runtime (s)	0.40	3.70

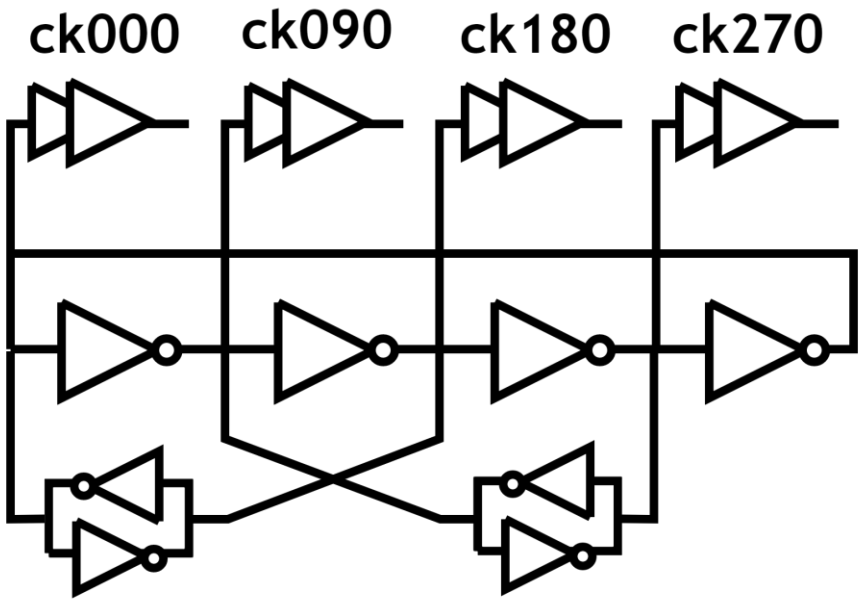
Stage		Insertion Delay (ps)			Rise/Fall Time (ps)		
		Manual	AutoCRAFT	RL	Manual	AutoCRAFT	RL
1	AVG STD	11.9 0.33	10.1 0.25	10.2 **0.23**	10.3 / 10.5 0.27 / 0.29	9.2 / 9.4 **0.22 / 0.24**	9.3 / 9.5 0.25 / 0.25
2	AVG STD	11.9 0.15	11.0 0.17	11.1 **0.15**	10.2 / 11.4 0.14 / 0.14	9.5 / 10.7 0.16 / 0.19	10.6 / 10.8 **0.12 / 0.12**
3	AVG STD	12.1 0.08	11.3 0.08	11.6 **0.06**	10.5 / 10.5 0.08 / 0.06	9.9 / 9.9 0.05 / 0.06	10.0 / 10.1 **0.02 / 0.00**
4	AVG STD	11.2 0.01	10.4 0.01	10.9 **0.00**	9.1 / 9.7 -	8.4 / 9.0 -	8.9 / 9.5 -
Tot.	AVG STD	77.7 0.53	72.7 0.42	74.1 **0.34**	- -	- -	- -

AutoCRAFT generates routing with shorter WL, while RL generates a more balanced layout

PTP path resistance difference reflects in STD of insertion delay and rise/fall time

4-Stage Ring Oscillator (OSC)

- Generates 10.3GHz complimentary in-phase (ck000/ck180) and quadrature-phase (ck090/ck270) clocks with a 750mV supply voltage
  - **21 cells, 16 nets, 102 pins**
  - **6 path-matching constraints, 14 paths**
  - **Manual placement**



Post-Layout Performance of OSC

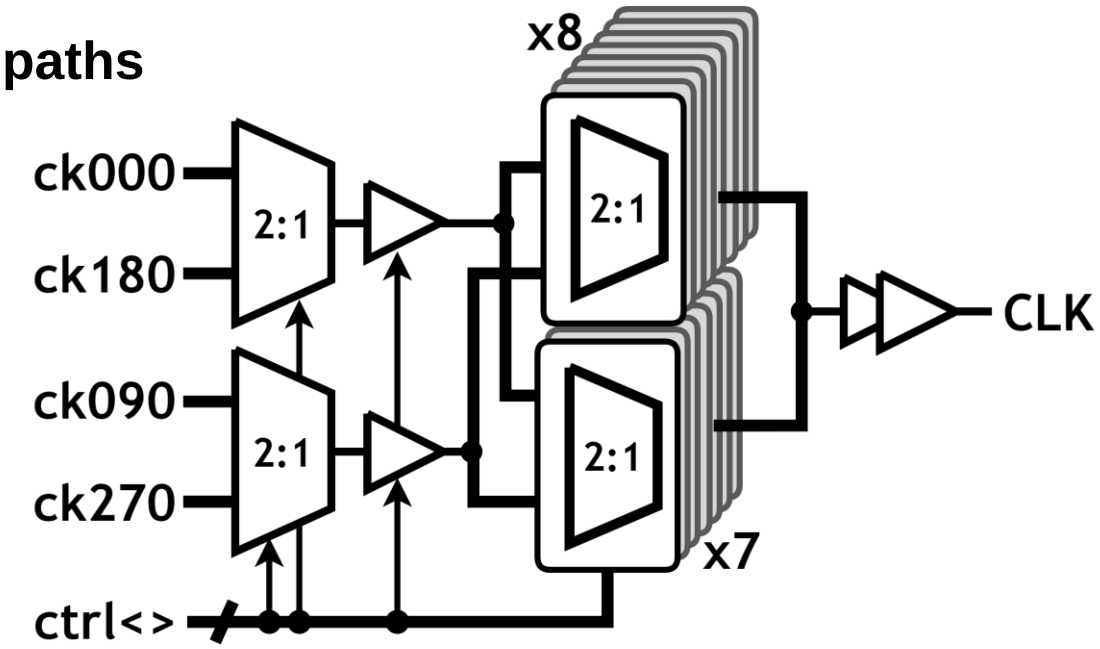
Metric	AutoCRAFT	RL
WL (μm)	21.3	21.0
VIA	88	90
DRV	5	0
PTP (Ω)	130.4	**23.9**
Runtime (s)	0.40	18.38

	Manual	AutoCRAFT	RL
Freq (GHz)	10.27	10.75	10.68
Duty Cycle			
ck000	49.0%	50.3%	50.2%
ck090	49.1%	50.3%	50.3%
ck180	48.9%	50.1%	50.2%
ck270	49.2%	50.1%	50.1%
Avg	49.1%	**50.2%**	**50.2%**
STD	0.10%	0.11%	**0.09%**
Quadrature Phase Distortion			
ck090	0.28%	-0.17%	0.15%
ck180	0.00%	-0.03%	-0.13%
ck270	0.33%	0.27%	0.05%
Avg	0.20%	0.16%	**0.11%**
STD	0.15%	0.10%	**0.04%**

RL-generated layout is more balanced

6-Bit Phase Interpolator (PI)

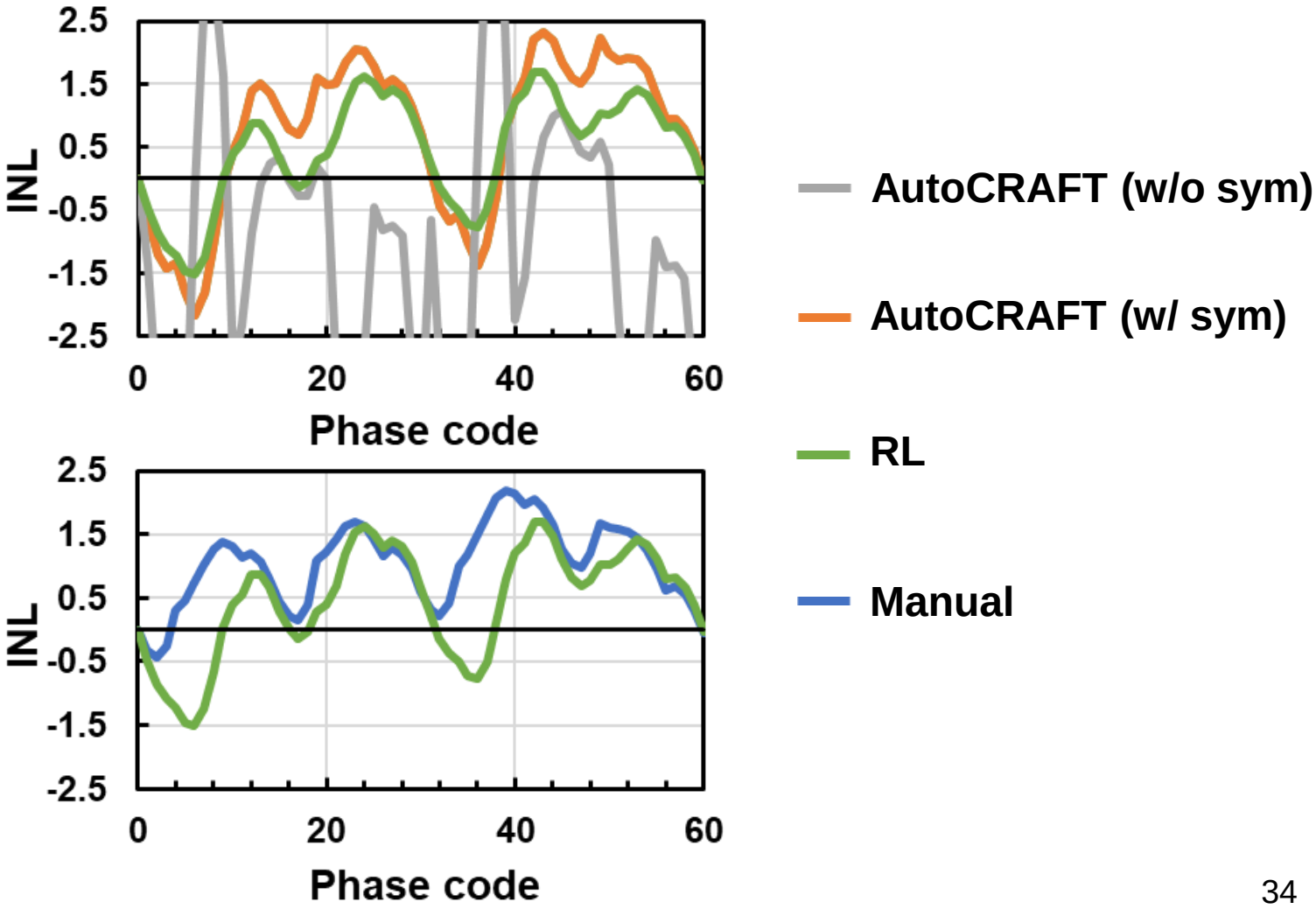
- The 6-bit phase interpolator adjusts the output clock phase in 1.33ps increments (60 phase steps) by interpolating between 4-phase input clocks (ck000, ck090, ck180, ck270) at 12.5GHz
  - **124 cells, 122 nets, 620 pins**
  - **19 path-matching constraints, 56 paths**
  - **Manual placement**



Post-Layout Performance of PI

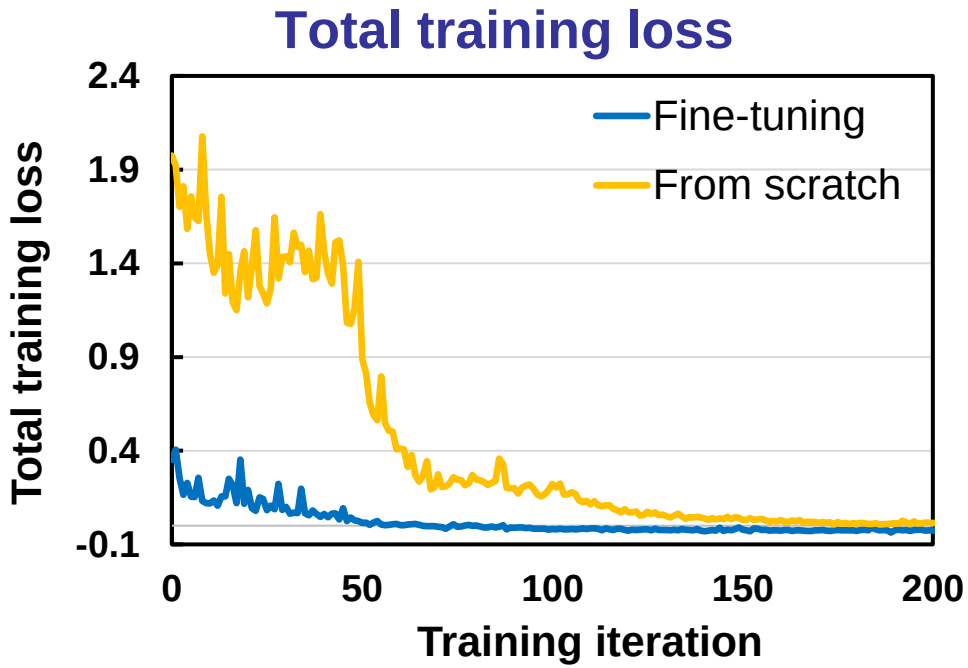
Metric	AutoCRAFT	RL
WL (μm)	380.7	385.8
VIA	779	790
DRV	1	0
PTP (Ω)	111.6	**38.2**
Runtime (s)	123.94	132.27

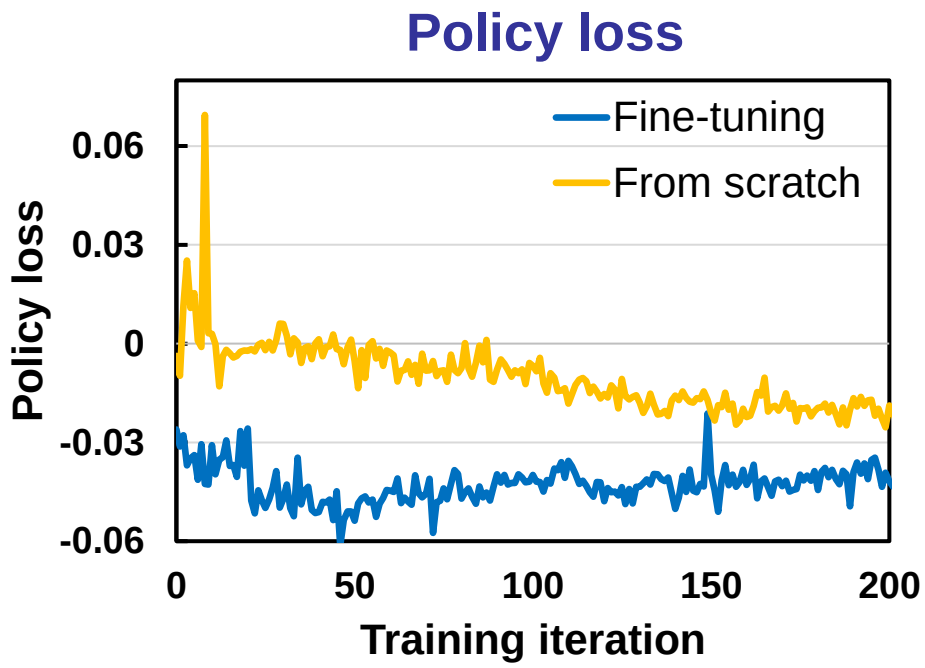
Integral non-linearity (INL) v.s. phase code



Policy Generalization

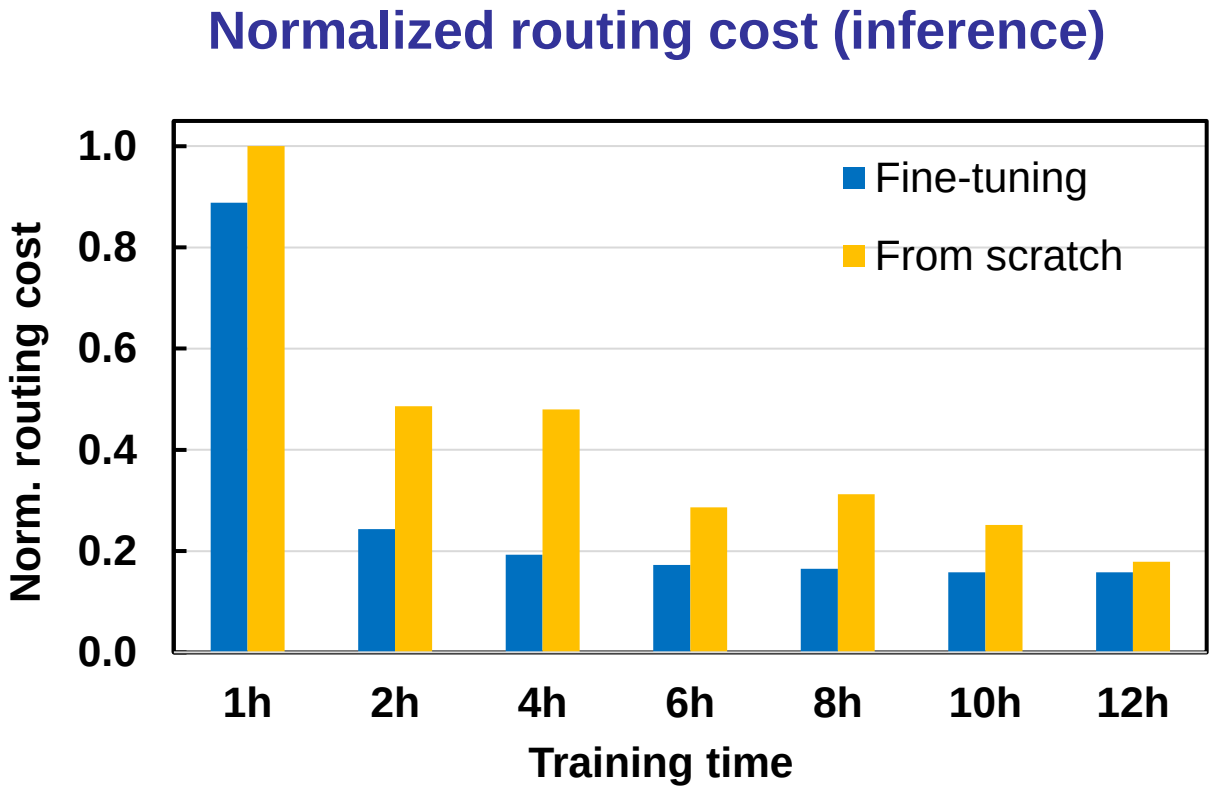
- Compare the training of PI with two different settings
  - Pre-train an RL policy using BUF and OSC, and do fine-tuning with PI
  - Train PI only from scratch





Policy Generalization

- Take the trained policies at different time stamps and integrate to the RL-based rip-up and re-route optimization procedure



Pre-trained models can be generalized to unseen circuits in some degree!!

Outline

- Background
- RL Guided Custom Detailed Routing Framework
- Experimental Results
- **Conclusion and Future Work**

Conclusion and Future Work

- **RL-based custom detailed routing framework**
  - Efficient heterogeneous graphs for routing solutions
  - Revised history cost accumulation scheme to encourage routing solution diversity
  - Path-matching constraint handling
  - Flexible framework that can extend to new objectives by adjusting the reward function
  - Generalizable RL policies for new/unseen circuits
- **Future direction**
  - Support more complicated custom routing strategies
    - Parallel routes, rings, …

Thank you!

Q&A