QPAR: A Plug-in Parallel Framework for Qubit Routing

Jiaqi Liu*, Runzhe Zhang*, Mingfei Zhang, Zhuo Song, Quan Chen, Jingwen Leng, Minyi Guo, Pu Pang†, Anbang Wu†

Shanghai Jiao Tong University

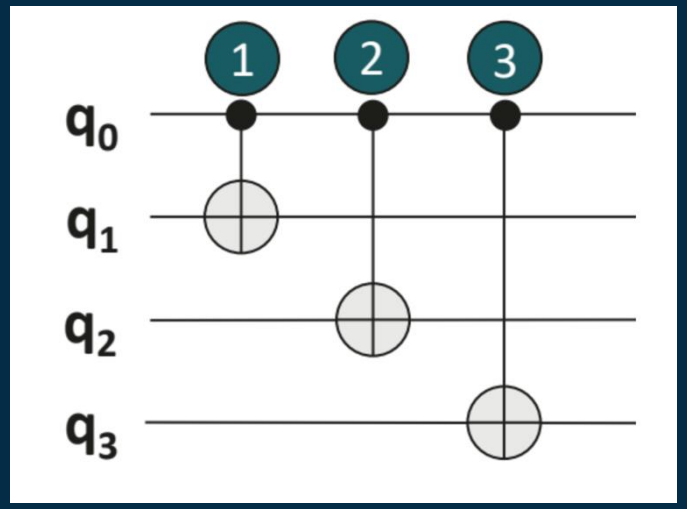


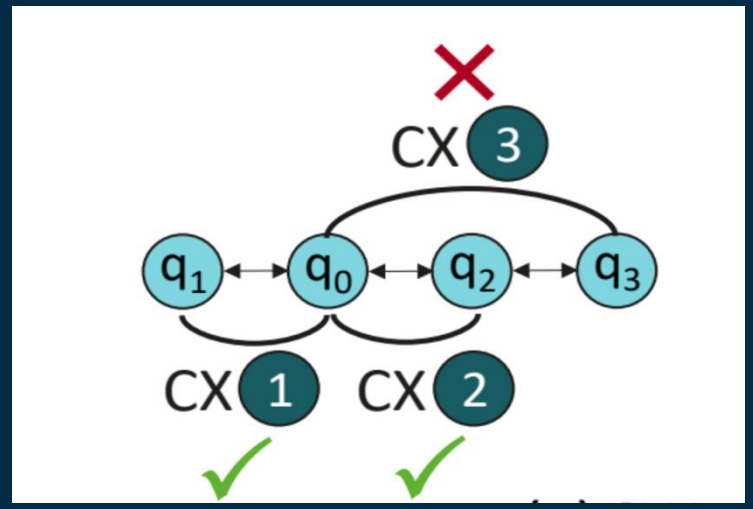
上海交通大學
SHANGHAI JIAO TONG UNIVERSITY



SC26 hpc unites | Chicago

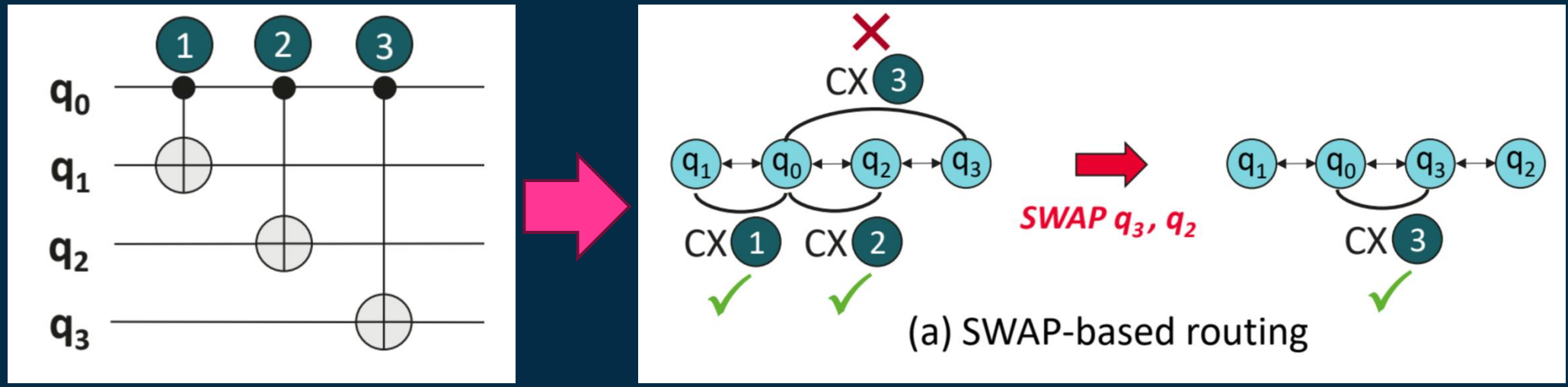
Background: Qubit Routing





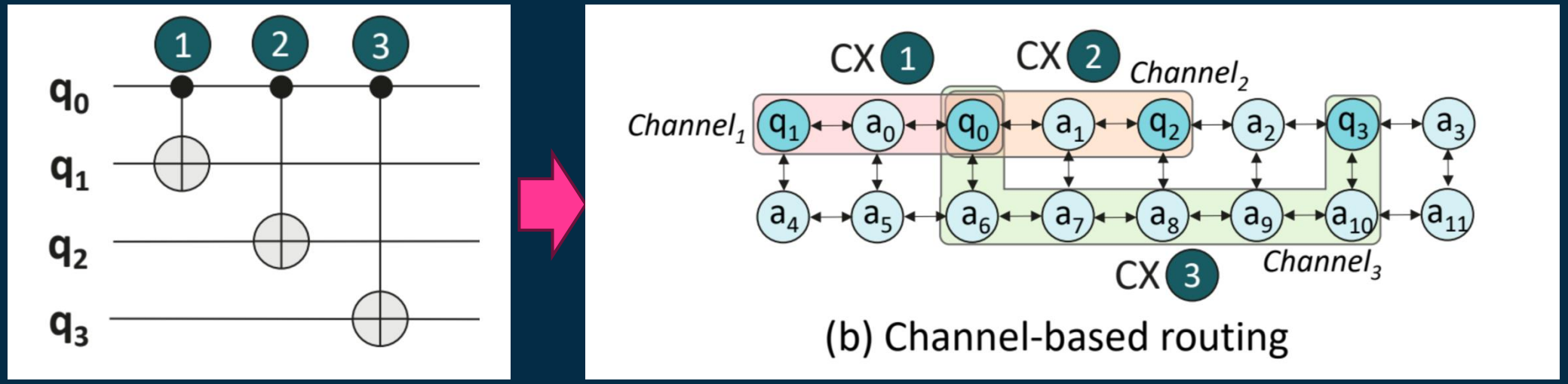
CX3 cannot run on this layout.

Background: Qubit Routing



One SWAP makes CX3 executable.

Background: Qubit Routing



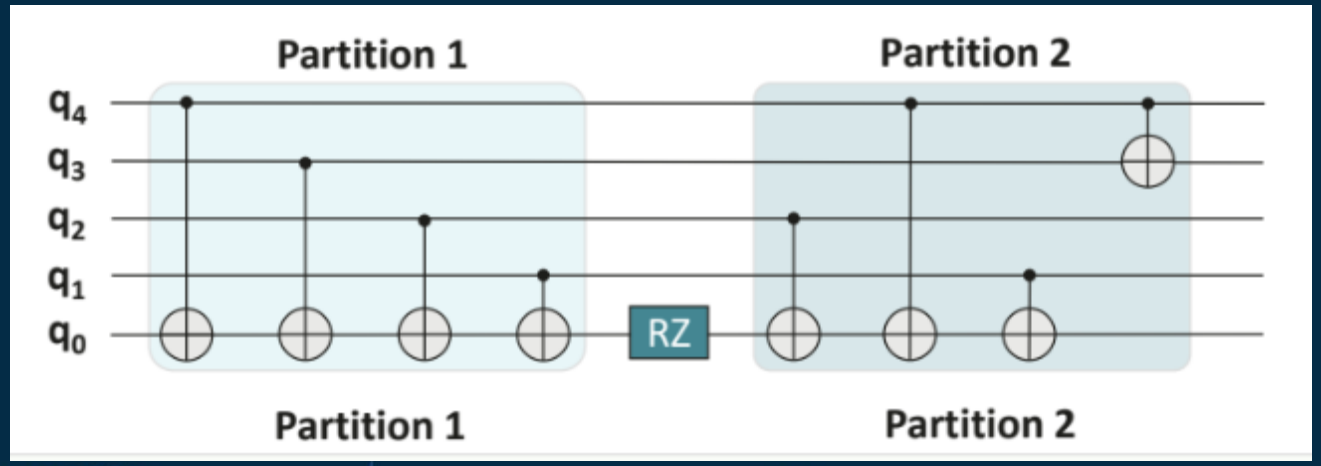
Ancilla channels enable remote CX without moving logical qubits.

Motivation: The Scalability Gap

- Routing quality matters: extra SWAPs or circuit space–time cost reduce the value of a compiled result.
- Existing routers often pursue better routing quality at high compilation cost.
- Large circuits turn routing scalability into a first-order concern.

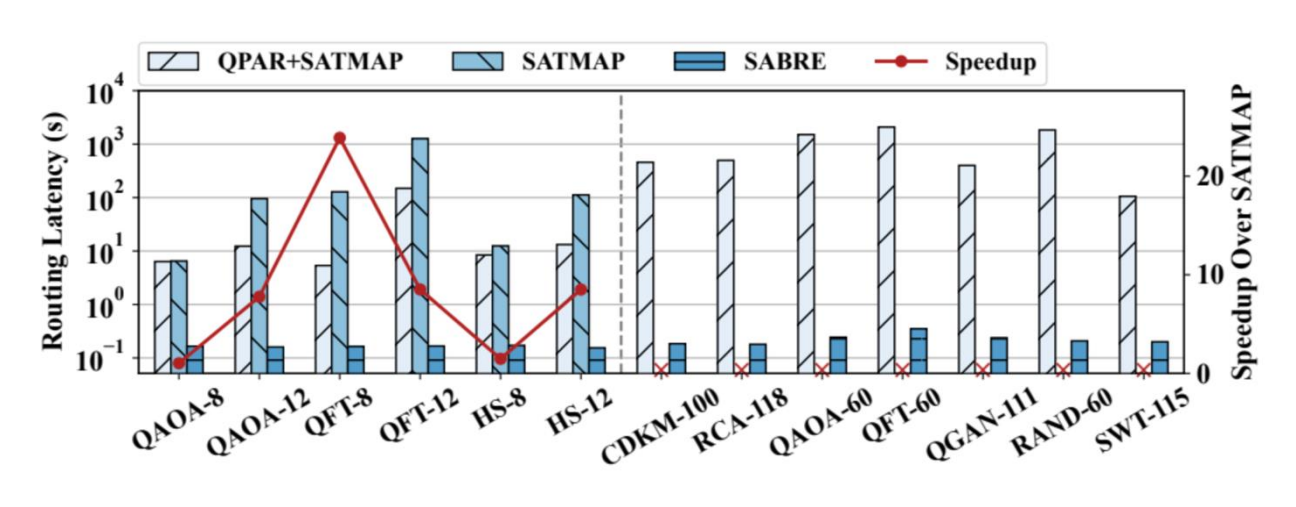
Investigation: Why Not Route in Parallel?

- A circuit is an ordered sequence of gates.
- The qubit layout after one region determines the starting layout of the next.
- Independent routing can break this mapping continuity.



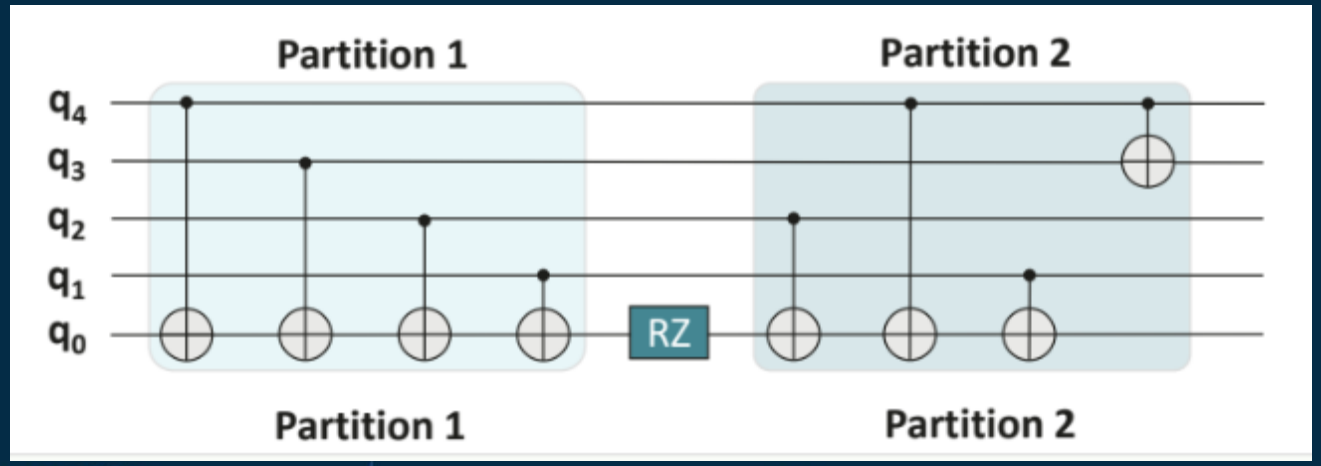
Challenges: Sequential Routing Is Expensive

- High-quality routing can become a compilation bottleneck.
- For several larger evaluated circuits, SATMAP produced no routing result within 24 hours.
- We need more throughput without trading away routing quality.



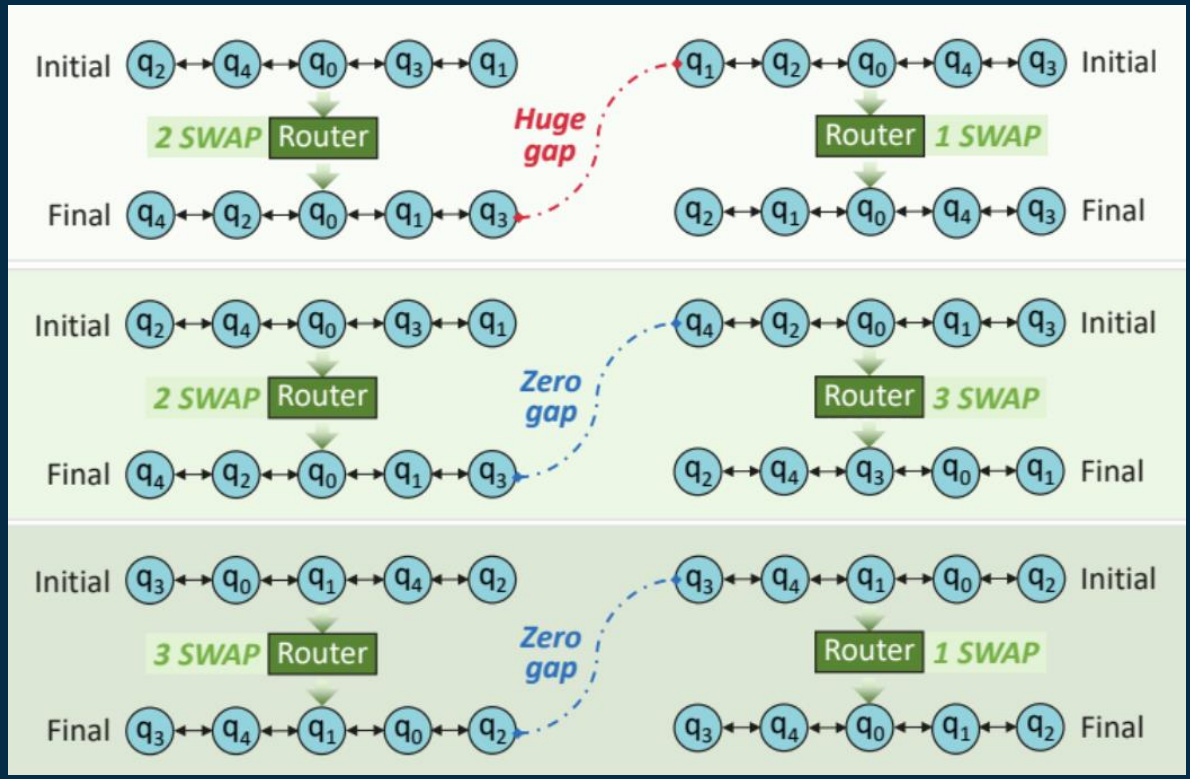
Challenges: The Sequential Dependency

- Where does the serial dependency live?
- In the qubit mapping passed from one gate block to the next.
- Relax that dependency, and sub-circuits can be routed concurrently.



Challenges: The Price of Parallelism

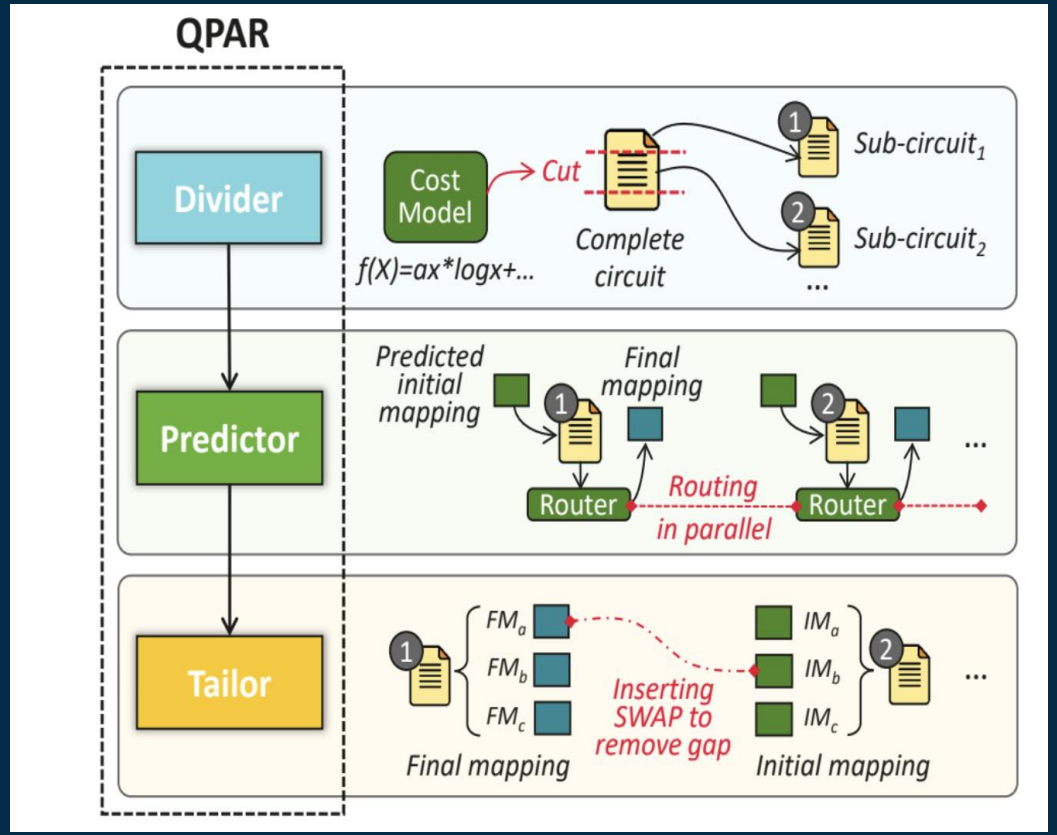
- Parallel routing produces an independent result for each sub-circuit.
- But FM(C_i) may differ from IM(C_{i+1}).
- Closing the gap can insert additional SWAPs.



Can we parallelize a sequential qubit router without losing routing effectiveness?

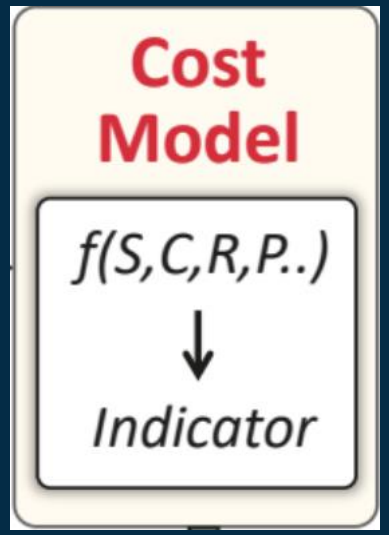
Insight: Relax, Then Rebuild Dependencies

- Routing is sequential because adjacent sub-circuits share a qubit-mapping dependency.
- Relax this dependency to route sub-circuits in parallel.
- Rebuild mapping continuity to obtain a valid complete routing.



Insight: More Partitions Are Not Always Better

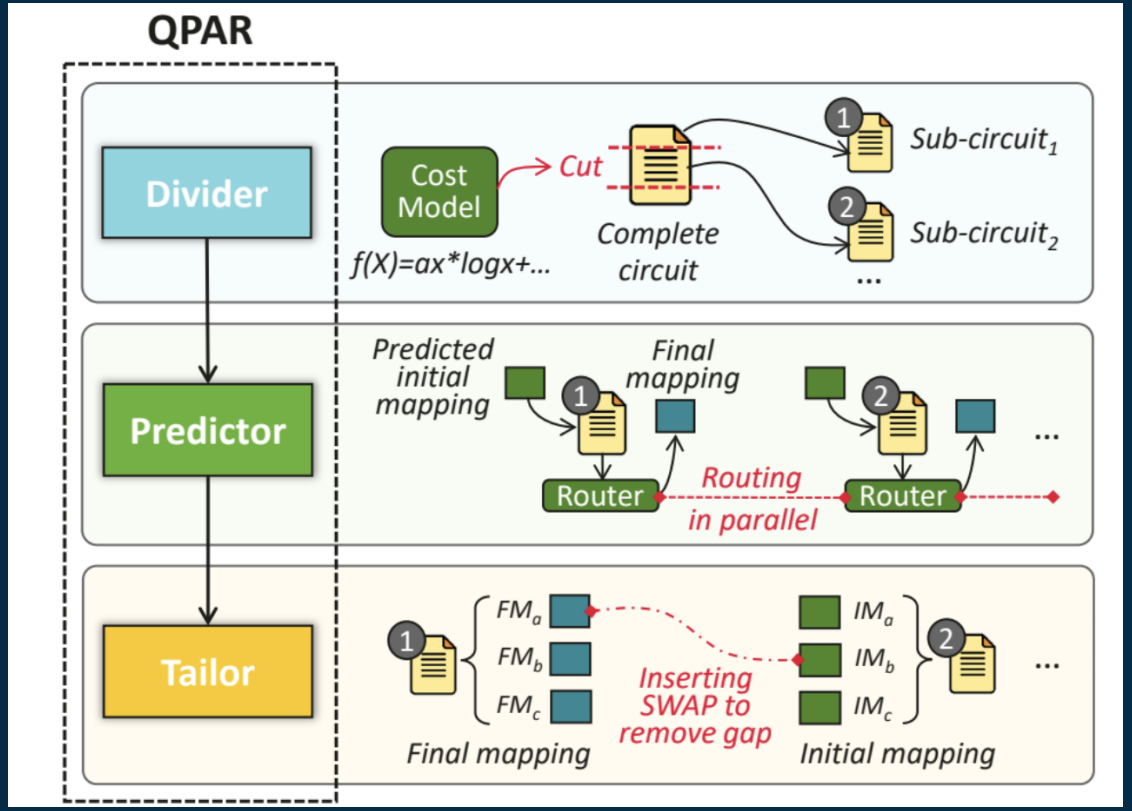
- More partitions can expose more routing parallelism.
- They also create more mapping boundaries and reconstruction work.
- Effective partitioning must balance speedup against routing quality.



Parallelism and reconstruction overhead

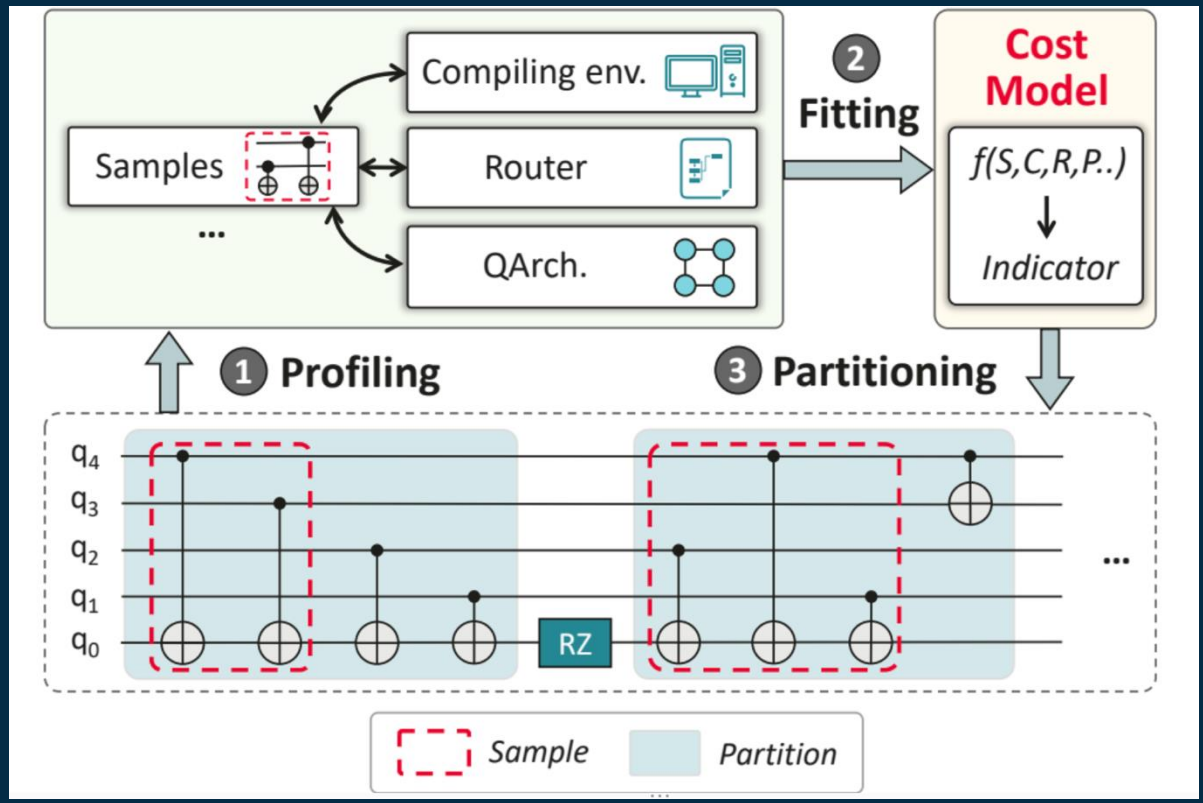
Our Method: QPAR Overview

- Relax sequential mapping dependencies to expose parallel routing.
- Rebuild those dependencies to recover a valid complete routing.
- QPAR plugs into existing routers without modifying them.



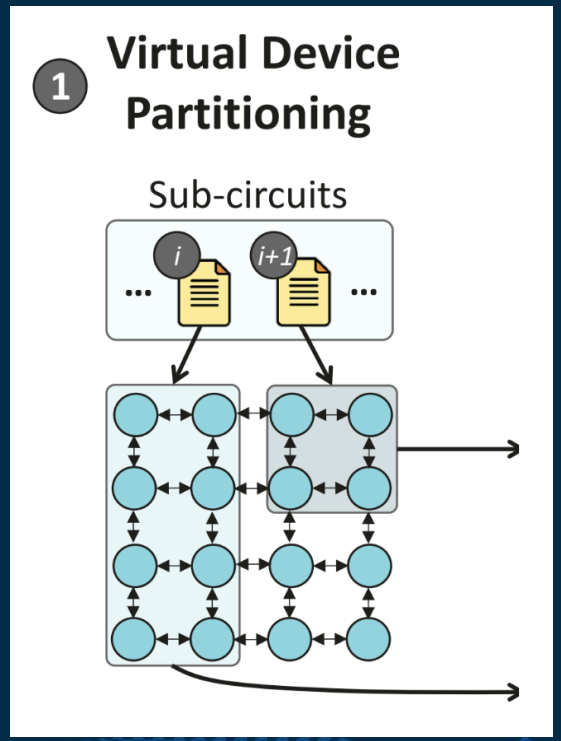
Our Method: Divider

- Profile representative circuit snippets on the target router and device.
- Estimate routing time and routing/reconstruction overhead for candidate cuts.
- Choose partitions that balance parallel speedup against dependency reconstruction overhead.



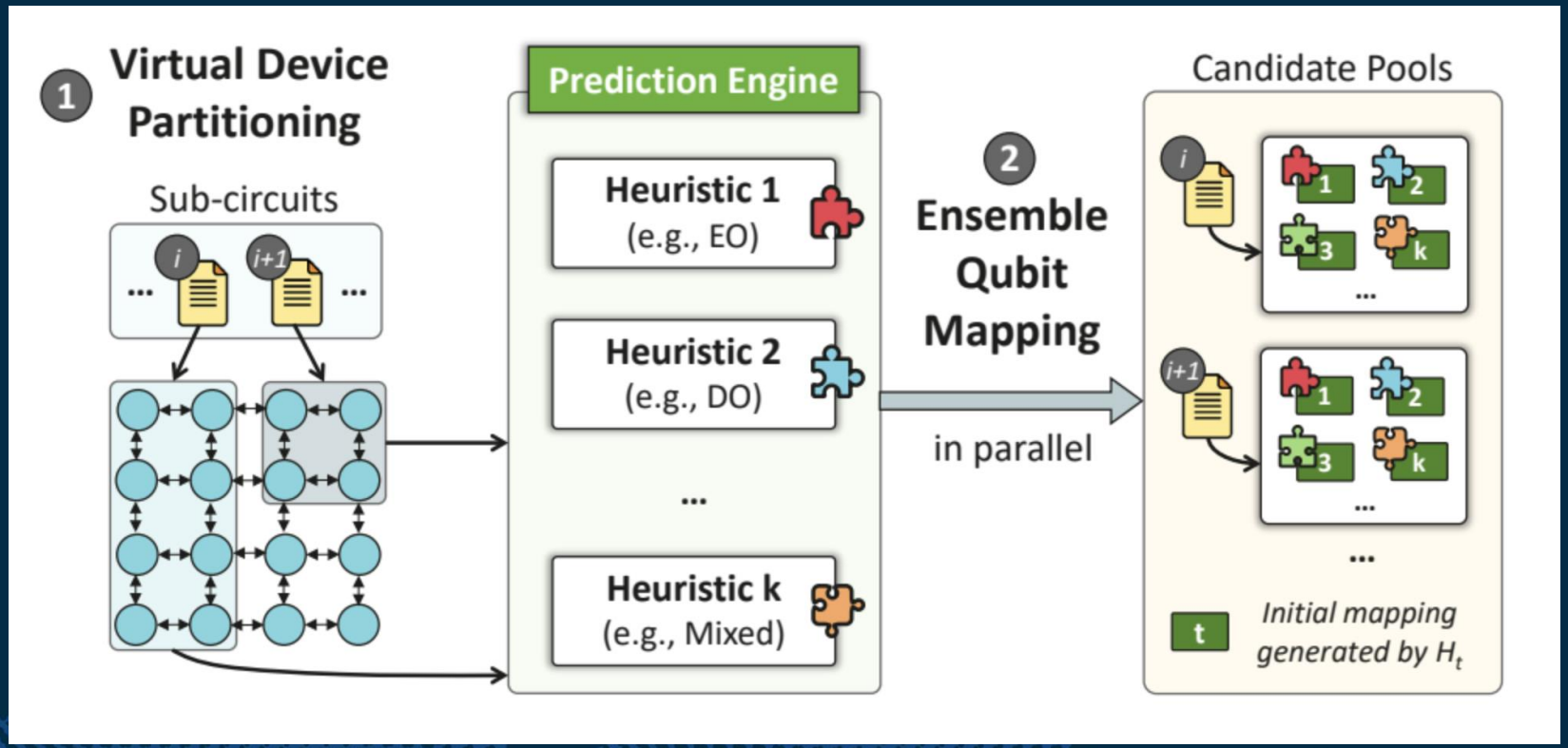
Our Method: Predictor / Locality

- Route each sub-circuit in a local device region.

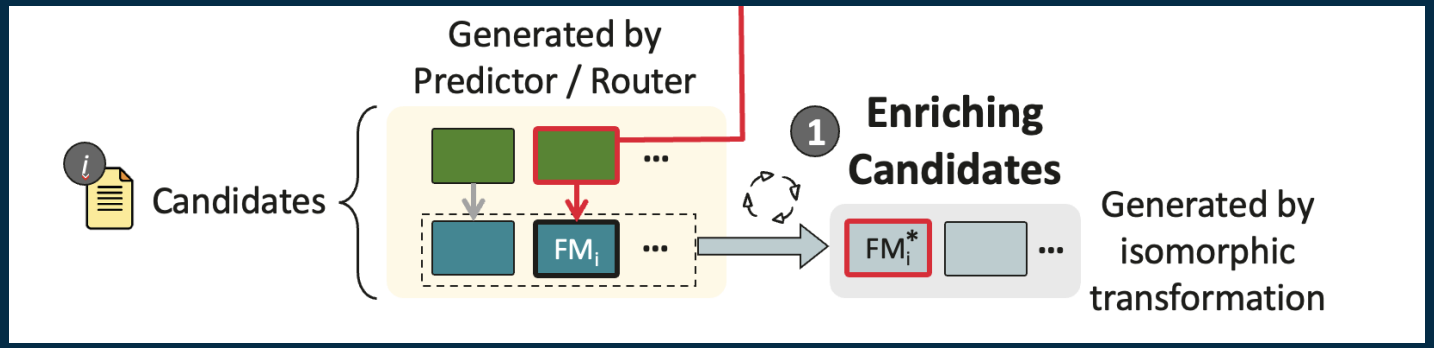


Our Method: Predictor / Mapping Ensemble

- Give each partition multiple mapping choices.

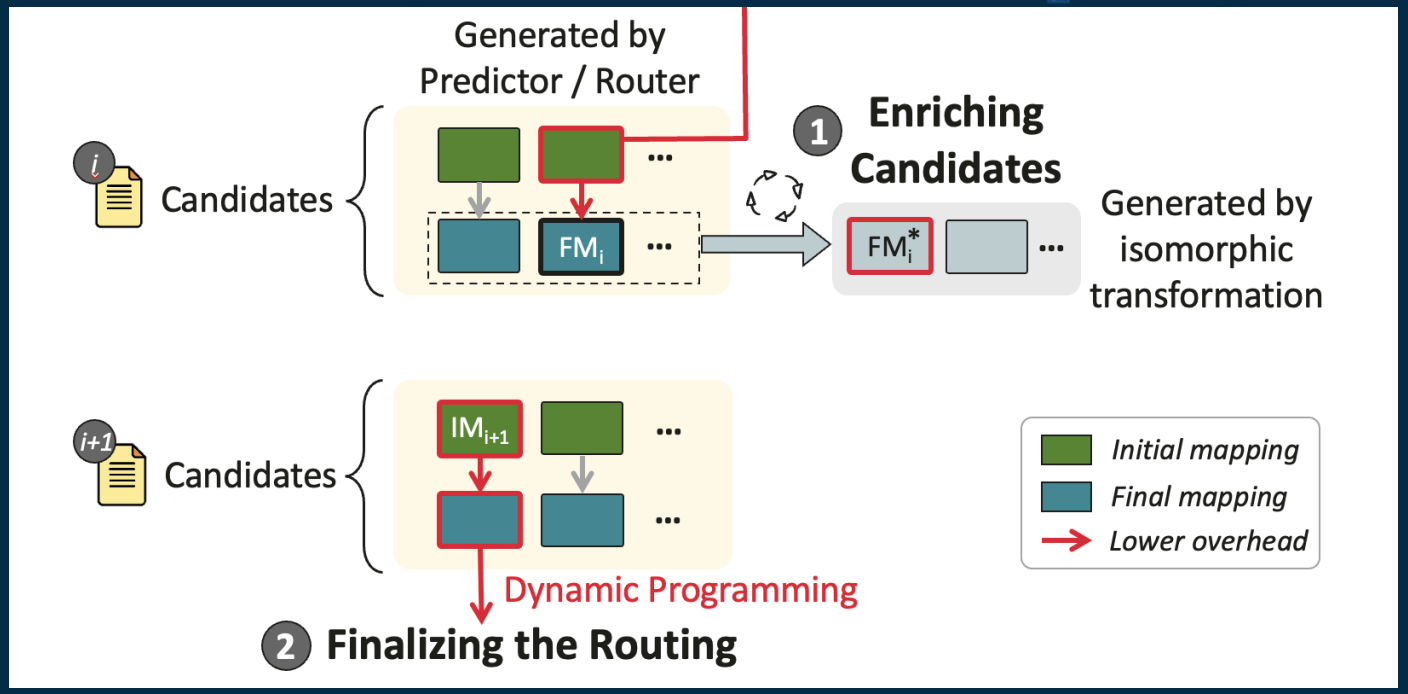


Our Method: Tailor / Enrich Candidates



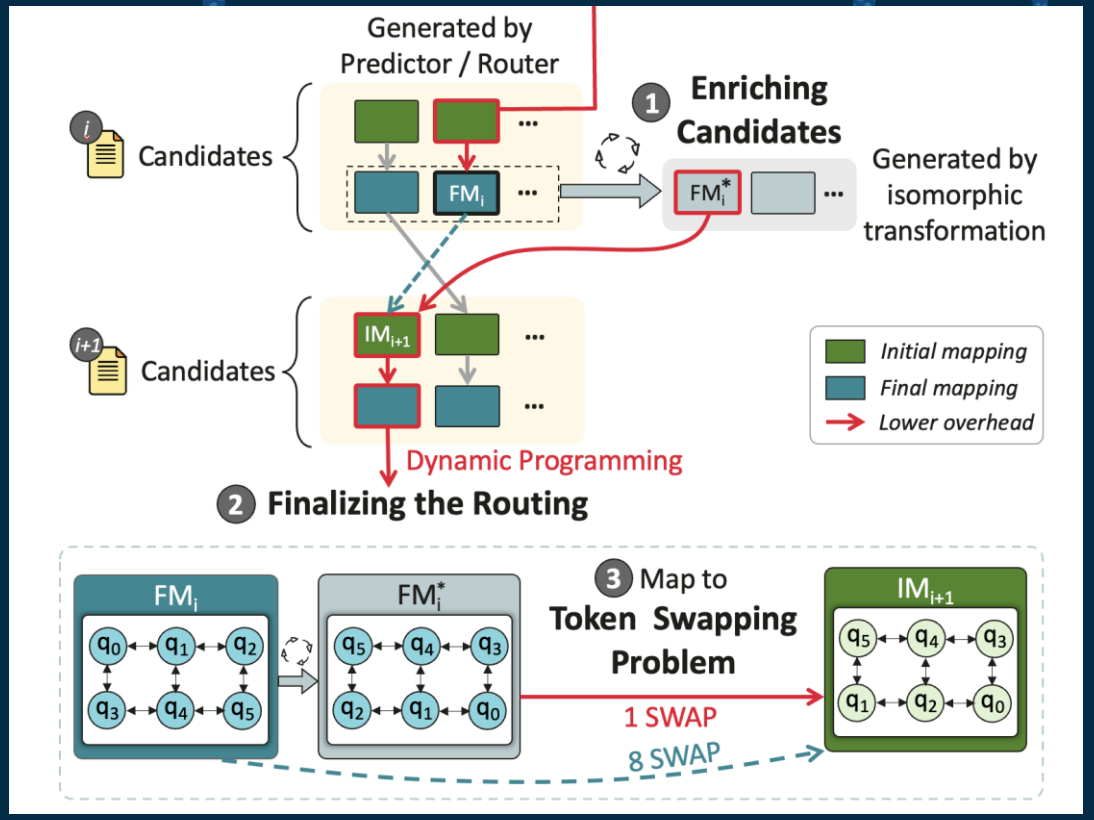
- Transform routed candidates to expand the choices.

Our Method: Tailor / Select Routing Candidates



- Use dynamic programming to select the best mapping sequence.

Our Method: Tailor / Restore Mapping Continuity



- Use token swapping to reconcile adjacent qubit mappings.

Evaluation: Setup

- Platform: 128 logical CPU cores; 512 GB RAM.
- Plug-in routers: SATMAP (SWAP-based) and EDPC (channel-based).
- Workloads: NISQ and FTQC circuits, including QAOA, QFT, Grover, and Shor.
- Metrics: routing latency, SWAP count, and circuit space–time overhead.
- Device topologies: square grid; IBM heavy hexagon for sensitivity analysis.

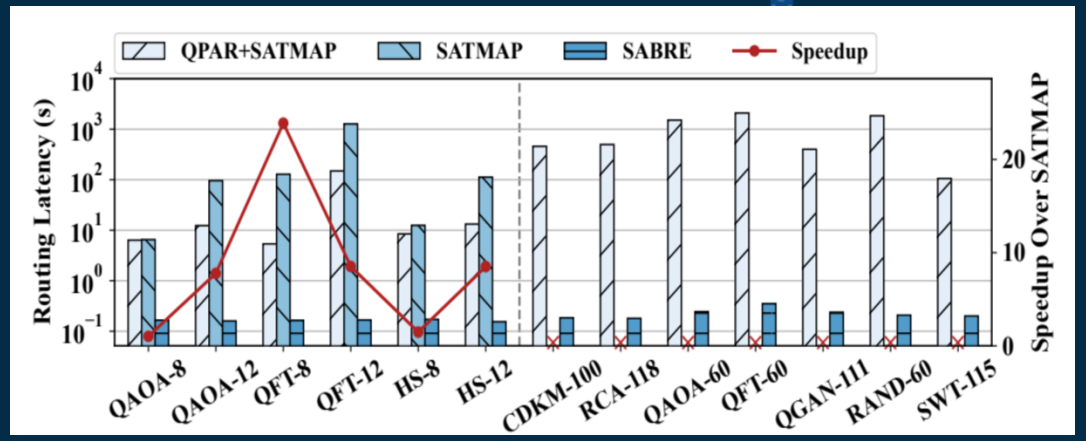
. USED TO EVALUATE SWAP-BASED ROUTING (NISQ+FTQC).

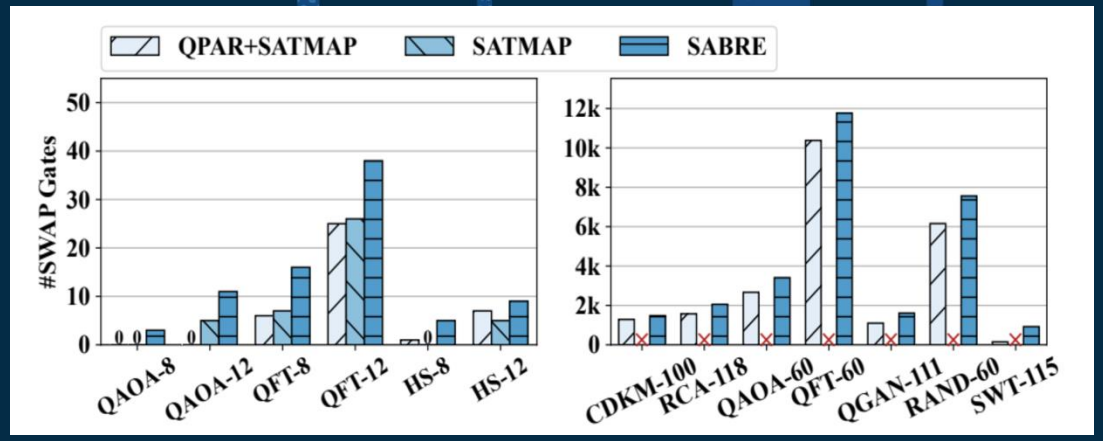
Circuit-#Qubits	#Gates	#CX	#1Q	Circuit-#Qubits	#Gates	#CX	#1Q
QAOA-8	46	20	26	HS-12	112	40	72
QAOA-12	84	40	44	CDKM-100	1569	785	784
QAOA-60	2910	1860	1050	RCA-118	2146	845	1301
QFT-8	48	40	8	QGAN-111	3052	872	2180
QFT-12	96	84	12	RAND-60	1900	1860	40
QFT-60	1920	1860	60	SWT-115	1716	456	1260
HS-8	104	40	64				

STATISTICS OF BENCHMARK PROGRAMS USED TO EVALUATE THE CHANNEL-BASED ROUTING ON FTQC.

Circuit-#Qubits	#Gates	#CX	#1Q	Circuit-#Qubits	#Gates	#CX	#1Q
HS-64	97,856	96,768	1,088	QWalk-64	33,344	32,256	1,088
HS-256	1,571,072	1,566,720	4,352	QWalk-256	526592	522,240	4,352
HS-1024	25,158,656	25,141,248	17,408	QWalk-1024	8397824	8,380,416	17,408
Grover-64	196,672	65,536	131,136	RAND-64	64,640	64,512	128
Grover-256	3,145,984	1,048,576	2,097,408	RAND-256	1,044,992	1,044,480	512
Grover-1024	50,332,672	16,777,216	33,555,456	RAND-1024	16,762,880	16,760,832	2,048
QAOA-64	66,624	64,512	2,112	Shor-64	107,473	101,312	6,161
QAOA-256	1,052,928	1,044,480	8,448	Shor-256	1,609,729	1,585,088	24,641
QAOA-1024	16,794,624	16,760,832	33,792	Shor-1024	25,313,473	25,214,912	98,561

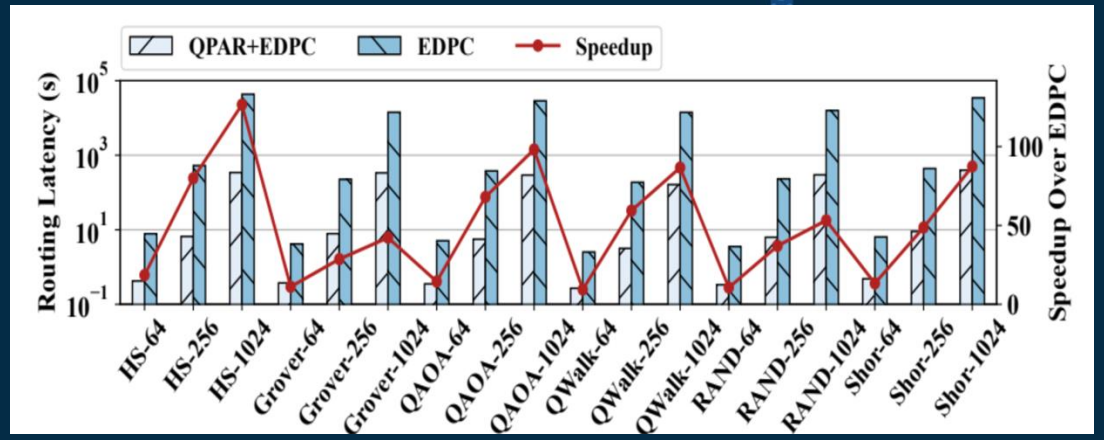
Evaluation: Swap-based Routing

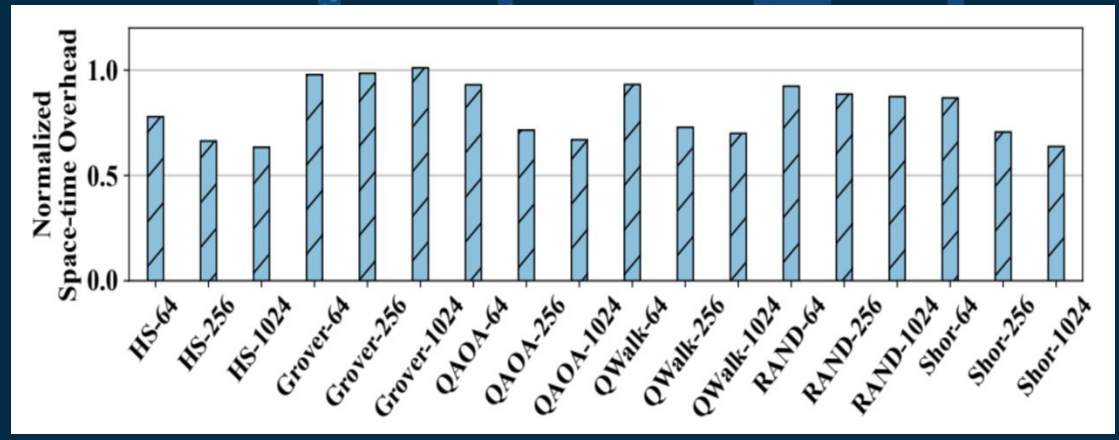




- QPAR + SATMAP achieves up to **23.9×** speedup on circuits SATMAP completes within a day.
- On larger tested circuits, QPAR + SATMAP finishes within **35** minutes while SATMAP does not finish within 24 hours.

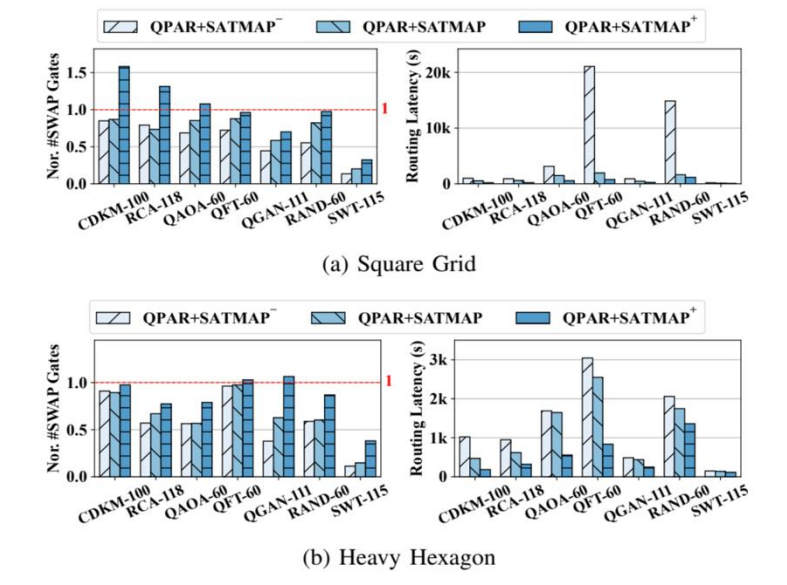
Evaluation: Channel-based Routing

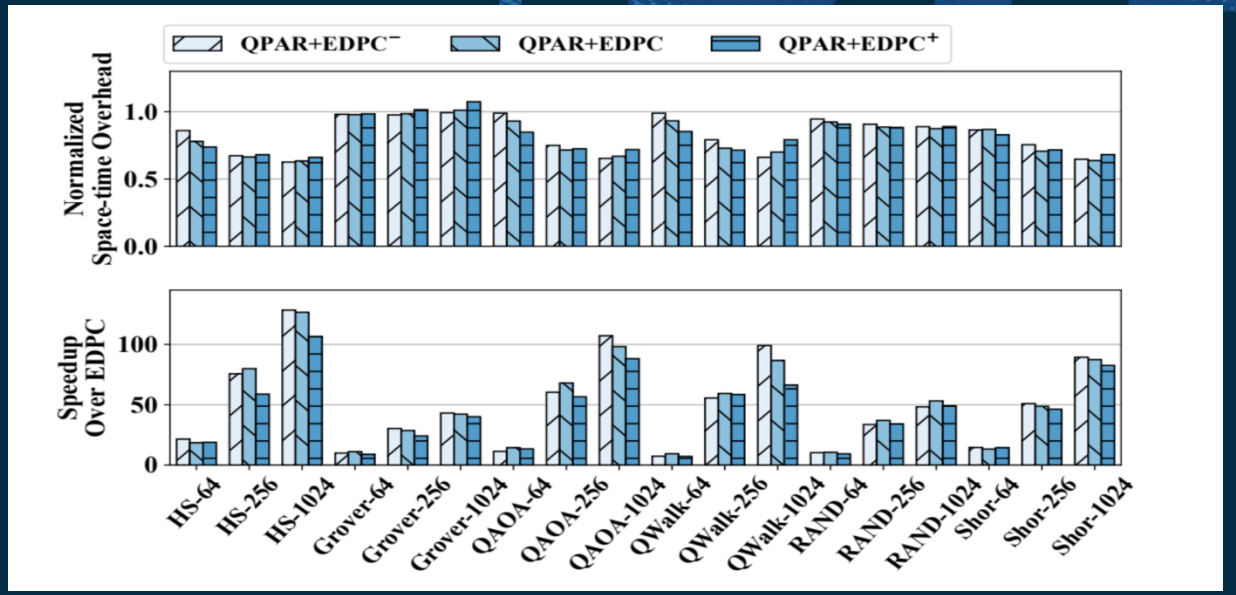




- QPAR + EDPC accelerates routing by up to **126.6×**.
- All tested circuits—including ones with about 50 million gates—finish within **6.6 minutes**.
- Circuit space–time overhead drops by up to 36.6%.

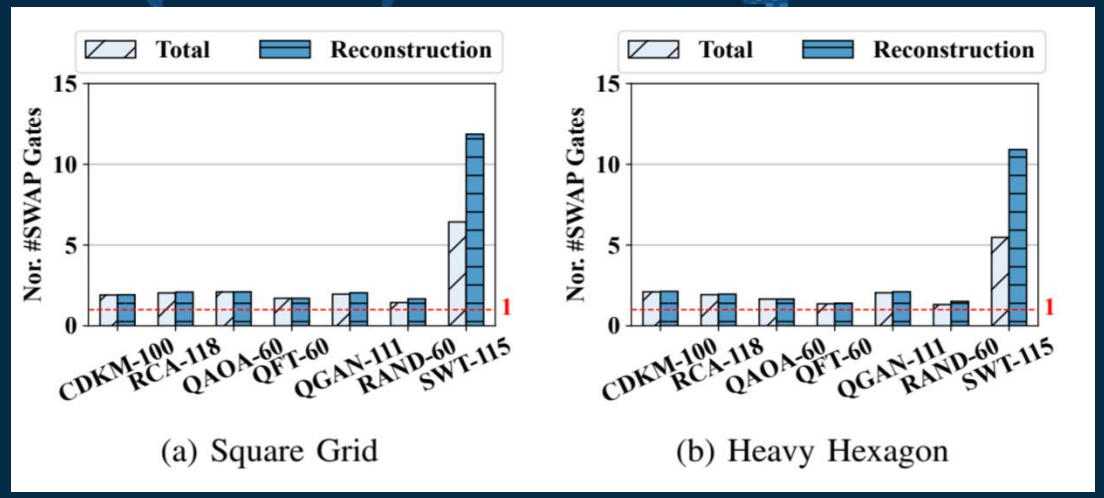
Evaluation: Divider Ablation

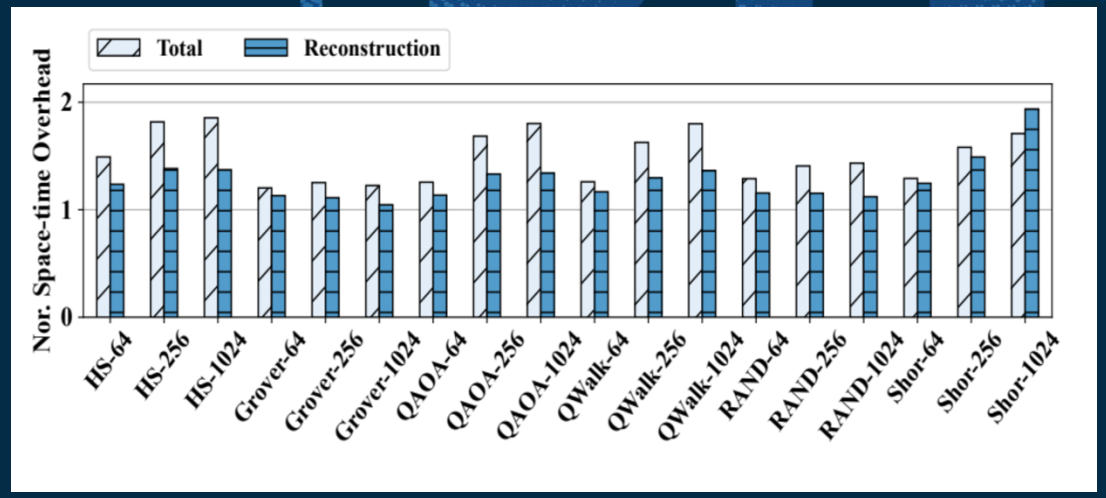




- More partitions: faster, but more SWAPs
- Fewer partitions: fewer SWAPs, but slower.

Evaluation: Predictor and Tailor Ablation





- Ablation: use one random mapping per sub-circuit and omit the tailor's mapping selection.
- Total routing overhead increases by up to **6.4×**.
- Reconstruction overhead increases by up to **11.9×**.

Conclusion

- ***Relax sequential mapping dependencies to route sub-circuits in parallel.***
- ***Rebuild mappings across boundaries to produce a valid, complete routing.***
- ***Plug into existing routers to accelerate compilation with low routing overhead.***

Conclusion

- ***Relax sequential mapping dependencies to route sub-circuits in parallel.***
- ***Rebuild mappings across boundaries to produce a valid, complete routing.***
- ***Plug into existing routers to accelerate compilation with low routing overhead.***

Q & A