IJCAI/2023 MACAO
The 32nd International Joint Conference on Artificial Intelligence
IJCAI 2023
19th–25th August 2023
Sheraton Grand Macao, Macao SAR

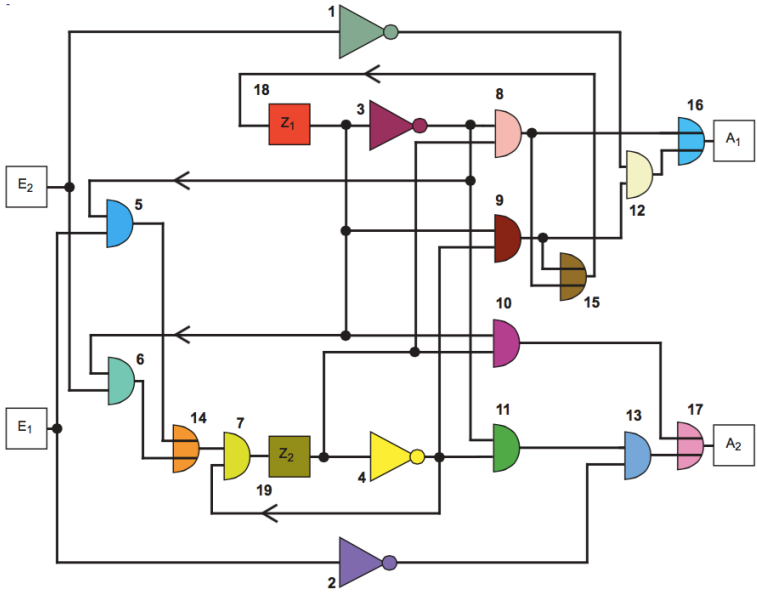
FastGR: Global Routing on CPU-GPU with Heterogeneous Task Graph Scheduler

Siting Liu[1,2], Yuan Pu[1], Peiyu Liao[1,2], Hongzhong Wu[3], Rui Zhang[3], Zhitang Chen[4], Wenlong Lv[4], Yibo Lin[2], **Bei Yu**[1]

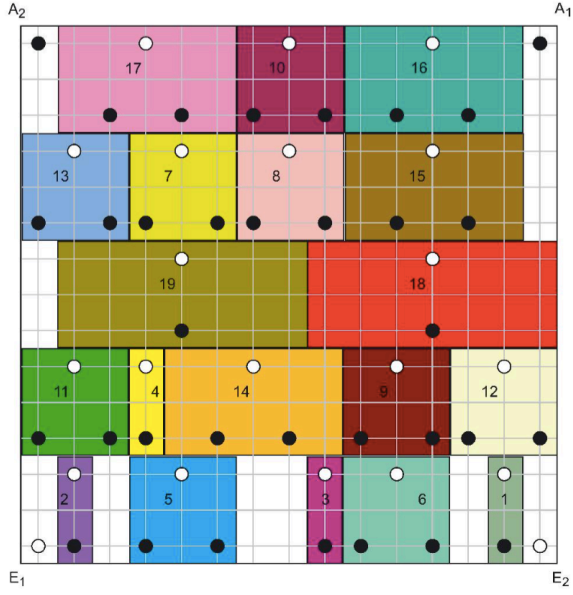
[1]The Chinese University of Hong Kong
[2]Peking University
[3]HiSilicon Technologies Co.
[4]Huawei Noah's Ark Lab

Introduction

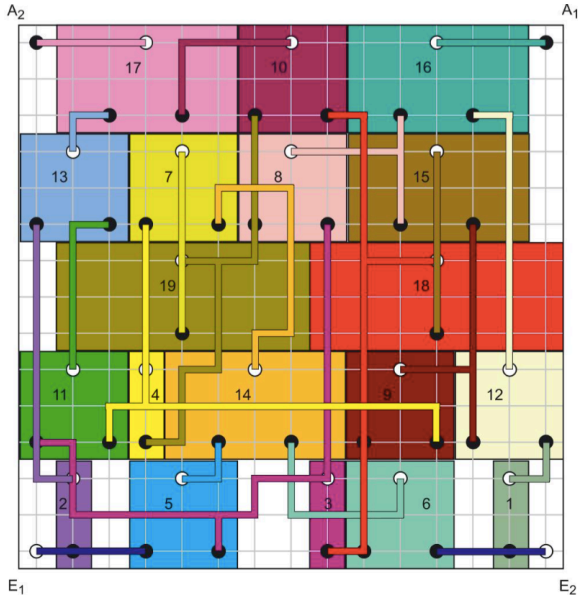
VLSI Placement & Routing: Toy Example

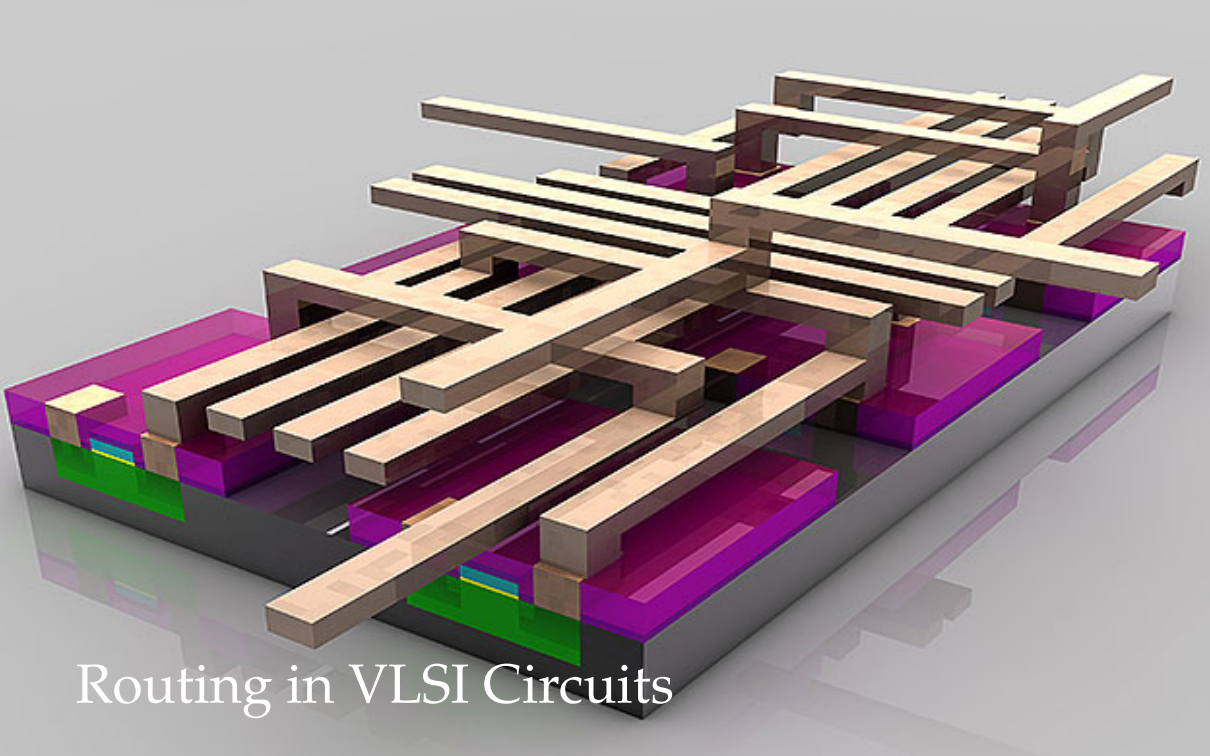


VLSI Placement & Routing: Toy Example

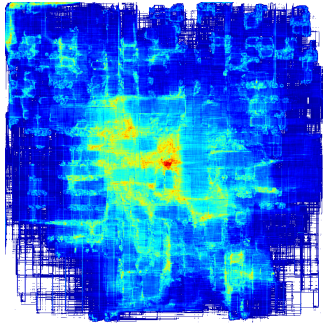


VLSI Placement & Routing: Toy Example



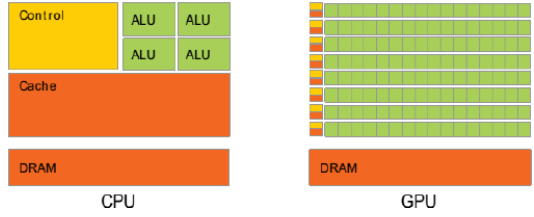

Routing in VLSI Circuits

Motivation



Routing solution sample.

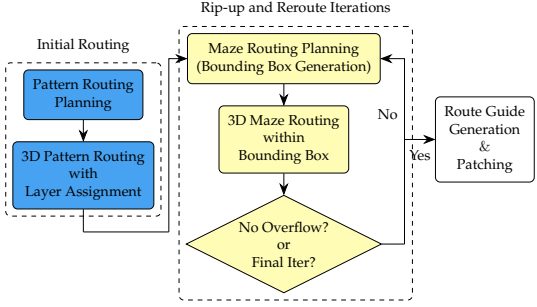
- 10+ metal layers.
- Millions of nets.
- Various objectives and constraints.

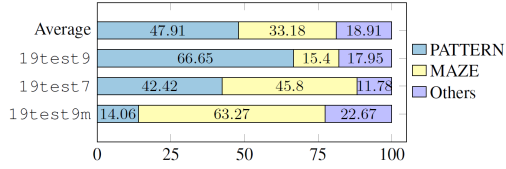


Heterogeneous architecture.

- CPU: Strong controller and ALUs.
- GPU: Grid-based computation resources: max 1024 threads per block.
- GPU: Cheap synchronization within blocks.

Motivation





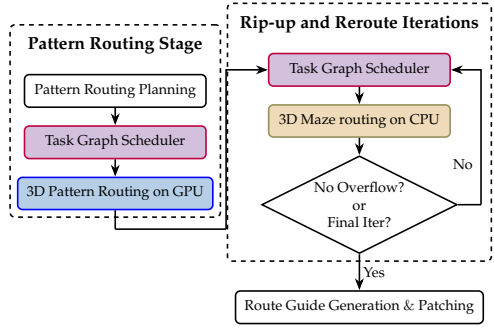
Runtime breakdown of a modern global router[1]; PATTERN means the pattern routing stage while MAZE means the maze routing stage.

[1]Liu, Jinwei, et al. "Cugr: Detailed-routability-driven 3d global routing with probabilistic resource model." 2020 57th ACM/IEEE DAC. IEEE, 2020.

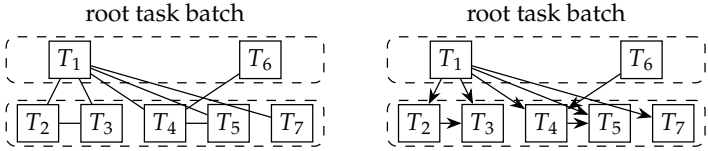
FastGR

Overall Framework

- A high-performance task graph scheduler to distribute CPU and GPU tasks for workload balancing and efficiency.
- A novel GPU-accelerated pattern routing framework that can route a batch of nets leveraging the massive parallelism on GPU.
- Two versions of FastGR are proposed including runtime-oriented version FastGR^{L} and quality-oriented version FastGR^{H}.



Task Graph Scheduler



All the tasks are split into two parts using the batch scheduler[2]:

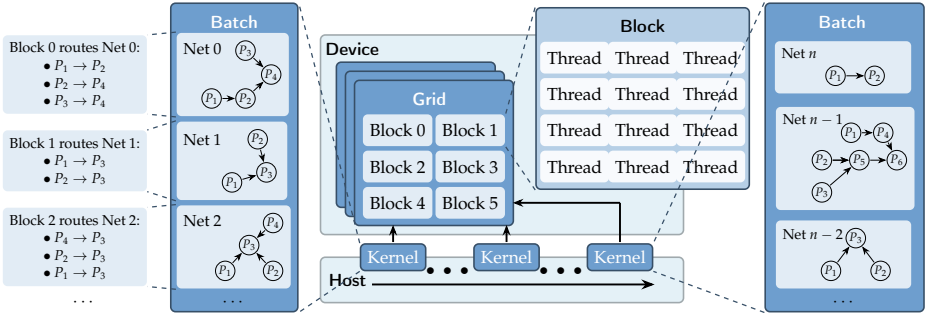
- Root task batch.
- Non-root task batch.

Execute direction

- *One task is part of the root batch, and the other is not.* The order is from the task in the root batch to the other.
- *Both the tasks are not in the root batch.* The order is from the task with a smaller task ID to the other.

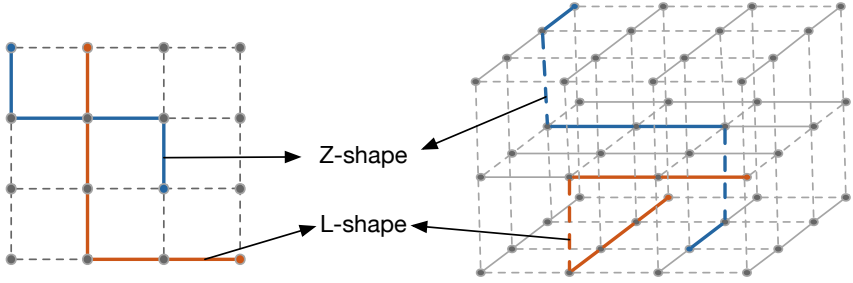
[2]G. Chen, C.-W. Pui, H. Li, and E. F. Young, "Dr. cu: Detailed routing bysparse grid graph and minimum-area-captured path search,"IEEE TCAD,vol. 39, no. 9, pp. 1902˘1915, 2019.

GPU-friendly Routing framework



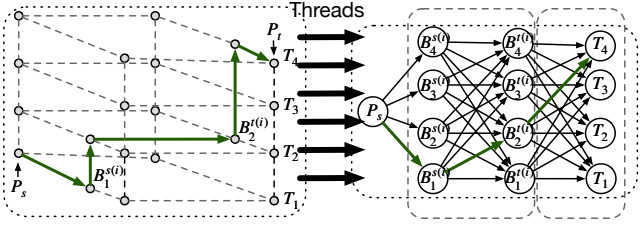
- Each kernel processes one batch of multi-pin nets.
- Different blocks route different multi-pin nets.
- The two-pin nets within one single multi-pin net are processed in sequential using dynamic programming-based routing.

Pattern routing



- Red paths: L-shape pattern routing solutions.
- Blue paths: Z-shape pattern routing solutions.

GPU-friendly 3D Pattern Routing

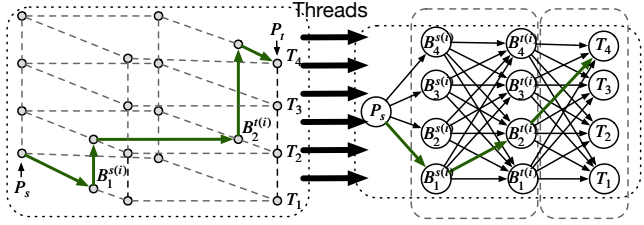


- the wire connecting the source point P_s to the source bend point $B_{l_s}^{s(i)}$;
- the vias to change routing metal layers from l_s to l_b and the wire connecting to the target bend point $B_{l_b}^{t(i)}$;
- the vias to change routing metal layers from l_b to l_t and the wire connecting to the target point T_{l_t}.

Path Cost

$$
\begin{aligned}
c(\mathcal{P}\{P_s, B_{l_s}^{s(i)}, B_{l_b}^{t(i)}, T_{l_t}\}) = & c_{\mathrm{w}}(P_s, B^{s(i)}, l_s) \\
& + c_{\mathrm{v}}(B^{s(i)}, l_s, l_b) + c_{\mathrm{w}}(B^{s(i)}, B^{t(i)}, l_b) \\
& + c_{\mathrm{v}}(B^{t(i)}, l_b, l_t) + c_{\mathrm{w}}(B^{t(i)}, T_{l_t}, l_t).
\end{aligned}
$$

Convert to Computation Graph Flow



3D Z-shape pattern routing for the i^{th} bend points pair $(B_s^{(i)}, B_t^{(i)})$

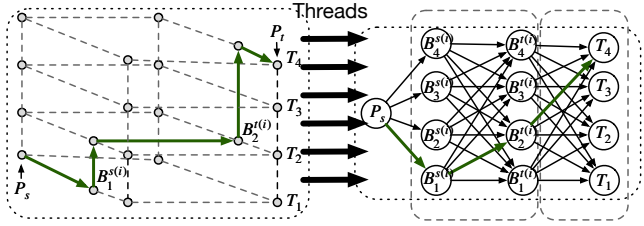
Computation Graph Weights

$$w_{l_s}^{(1)} = c_{\text{bc}}(P_s, l_s) + c_{\text{w}}(P_s, B^{s(i)}, l_s),\ 0 < l_s \leq L.$$

$$w_{l_s,l_b}^{(2)} = c_{\text{v}}(B^{s(i)}, l_s, l_b) + c_{\text{w}}(B^{s(i)}, B^{t(i)}, l_b),\ 0 < l_s, l_b \leq L.$$

$$w_{l_b,l_t}^{(3)} = c_{\text{v}}(B^{t(i)}, l_b, l_t) + c_{\text{w}}(B^{t(i)}, T_{l_t}, l_t),\ 0 < l_b, l_t \leq L.$$

Convert to Computation Graph Flow



3D Z-shape pattern routing for the i^{th} bend points pair $(B_s^{(i)}, B_t^{(i)})$

Computation Graph Flow

$$c^{*(i)}(P_s, P_t, l_t) = \min_{0<l_s,l_b\leq L} \left\{ w_{l_s}^{(1)} + w_{l_s,l_b}^{(2)} + w_{l_b,l_t}^{(3)} \right\}.$$

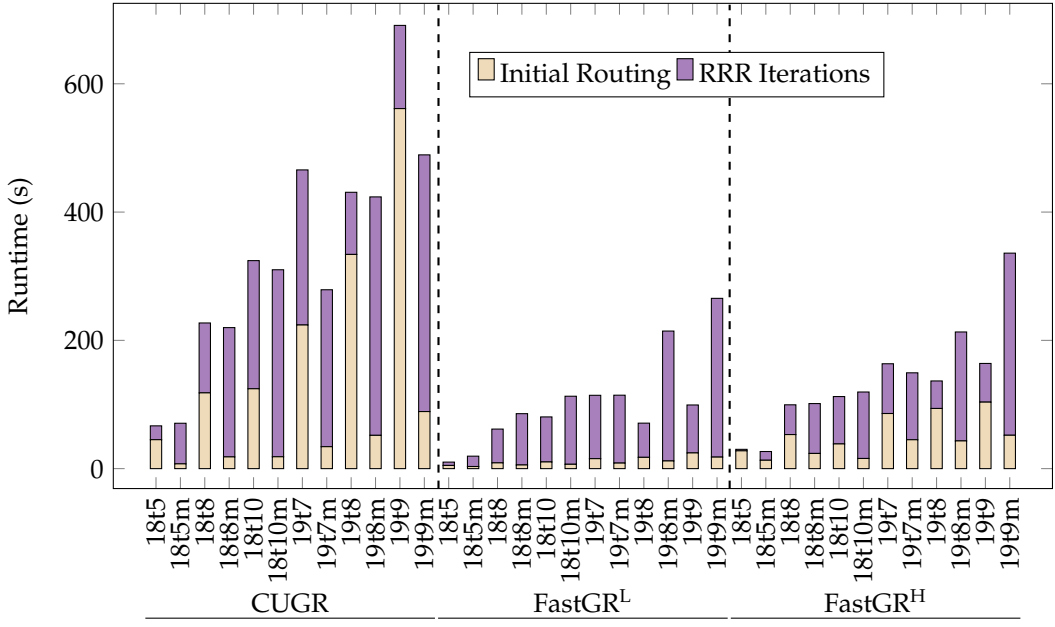
Results

Experimental Setting

- Device: 1 NVIDIA GeForce RTX 3090 GPU.
- Benchmark: ICCAD 2019 benchmarks.
- Sorting Strategy: Ascending bounding box half permeter.
- Pattern routing stage: GPU-friendly pattern routing framework + Task graph scheduler.
- Rip-up and reroute stage: Task graph scheduler.

- FastGR^{L}: GPU-friendly L-shape pattern routing.
- FastGR^{H}: GPU-friendly Hybrid-shape pattern routing.

Runtime Comparison



Solution Quality Results

Table: Routing solution quality results

Bench	Score		Wirelength		# Vias		# Shorts		
	FastGR^{L}	FastGR^{H}	FastGR^{L}	FastGR^{H}	FastGR^{L}	FastGR^{H}	FastGR^{L}	FastGR^{H}	Improved (%)
18test5	16919500	**16880800**	26988200	**26915900**	856362	**855723**	0	0	
18test5m	18774900	**18705100**	27942700	**27799600**	**802385**	809459	3188	**3135**	1.662
18test8	40881100	**40825200**	64359900	**64211700**	**2174240**	2179840	8.5	**0**	100.000
18test8m	42897000	**42728700**	64554600	**64441200**	**1948740**	1957020	5649.5	**5360**	5.124
18test10	42592100	**42575900**	66718200	**66677100**	**2308250**	2309320	0	0	
18test10m	44579100	**44275300**	71071700	**71014000**	**2059820**	2067080	1608	**1000**	37.811
19test7	71887200	**71763600**	118744000	**118495000**	**3128750**	3129020	0	0	
19test7m	67923300	**67639200**	107115000	**106577000**	**3080660**	3084060	4086.5	**4028.5**	1.419
19test8	113929000	**113844000**	181854000	**181687000**	5750370	**5750260**	0	0	
19test8m	**114362000**	114545000	177638000	**177284000**	**5612590**	5623080	**6185.5**	6822	-10.290
19test9	175523000	**175367000**	274106000	**273884000**	9603400	**9603200**	112.5	**23.5**	79.111
19test9m	173339000	**173002000**	267786000	**267304000**	**9359980**	9375940	4013	**3692**	7.999
Average									27.855

THANK YOU!