Starfish: An Efficient P&R Co-Optimization Engine with A*-based Partial Rerouting

Fangzhou Wang, Lixin Liu, Jingsong Chen, Jinwei Liu, Xinshi Zang, and Martin D.F. Wong

CSE Dept., The Chinese University of Hong Kong

Nov. 03, 2021

Outline

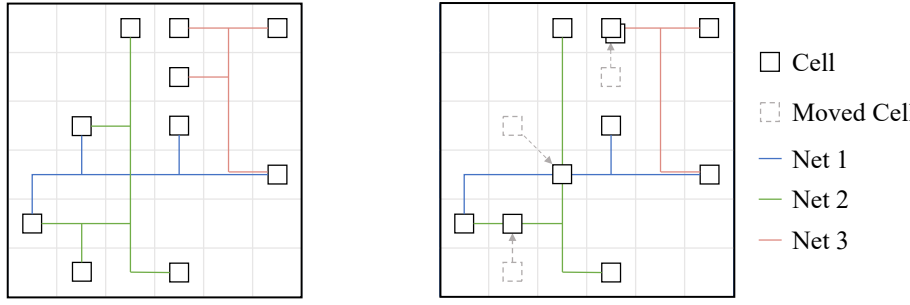
Outline

Placement and Routing (P&R)

A common two-stage flow

- Placement and Routing as 2 sub-problems
- **Placement:** decide the locations of circuit components
  - Objective: focuses on optimizing the half-perimeter wire length (HPWL) under cell/pin density constraints
- **Routing:** build the connection between cells
  - Objective: minimize the routed wire length and the number of overflows
- Pros: Reduction in design complexity
- Cons: Misalignment between objectives can degrade the final solution quality

Misalignment between Placement and Routing



(a) Before Routing with Cell Movement (b) After Routing with Cell Movement

Illustration for routing with cell movement. Both (a) and (b) have the same HPWL, but the routed wire length in (b) is shorter.

Routing with Cell Movement (ICCAD 2020 Contest)

Input

- 3D Design layout (L $\times$ R $\times$ C)
- Netlist and min-routing-layer for each net
- Initial placement and routing result
- Max-cell-move

Goals

Generate a set of **placement** and **routing** (P&R) results that

- minimize total routed wire length (that is, maximize the wire length reduction)
- comply with hard constraints:
  - max-cell-move
  - min-routing-layer
  - connectivity
  - preferred routing direction
  - overflow-free

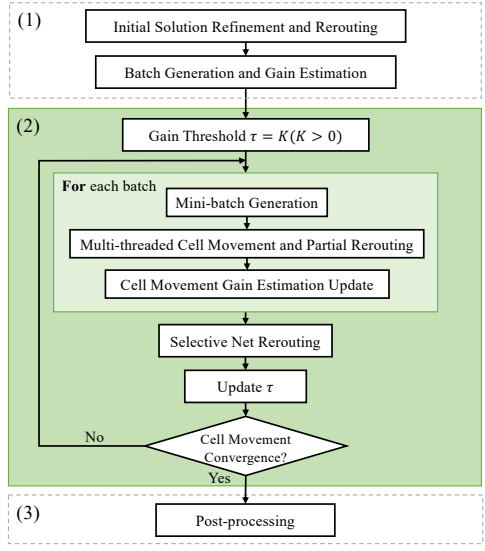
Outline

Overall Flow

(1) Pre-processing
- Initial solution refinement
- Congestion-driven rerouting

(2) Multi-threaded cell movement
- Lookup table-based gain estimation
- A*-based partial rerouting
- Selective net rerouting: detours removal
- Gain threshold τ for implicit cell ordering

(3) Post-processing
- Greedy cell put-back
- Wire length-driven rerouting



Overall flow.

Outline

Comparison with the Top-3 Winners

Table: Experimental Results on the ICCAD 2020 Benchmark[1]

Benchmarks		1st Place Team		2nd Place Team		3rd Place Team		Ours (Starfish)		
Case ID	TWL_{input}	Score	RT (s)	Score	RT (s)	Score	RT (s)	Score	RT (s)	R_{diff}
case3	32600	11425	34	11428	4	11557	111	**11610**	2	0.356
case4	4680681	2046811	2221	2048105	804	2037598	3441	**2064790**	260	0.441
case5	1763627	695219	903	685173	183	682963	1213	**699626**	111	0.397
case6	7188481	2721274	3171	2687926	2217	2656320	3511	**2737028**	684	0.381
case3B	29748	11237	31	11073	4	11289	206	**11327**	1	0.381
case4B	4886698	2182574	2299	2180172	786	2167411	3371	**2200820**	284	0.450
case5B	1721530	664347	886	654797	237	654183	1226	**668351**	103	0.388
case6B	7340802	2748097	3406	2722222	2801	2668052	3514	**2785061**	932	0.379
Avg. ratio	-	1.000	1.000	0.992	0.368	0.990	2.224	1.009	0.133	0.397

* The score and runtime statistics of the top-3 winners are provided by the contest organizer with Intel Xeon E7-4820 CPU (2.00 GHz, 8 cores). For us, all experiments are run on a Linux machine with 2.90 GHz Intel Xeon CPU and 8 threaded enabled.

[1] Kai-Shun Hu et al. (2020). "ICCAD-2020 CAD contest in Routing with Cell Movement". In: *Proc. ICCAD*.

Thank You!

References

Hu, Kai-Shun, Ming-Jen Yang, Tao-Chun Yu, and Guan-Chuen Chen (2020). "ICCAD-2020 CAD contest in Routing with Cell Movement". In: *Proc. ICCAD*.