Two-stage PCB Routing Using Polygon-based Dynamic Partitioning and MCTS

Youbiao He
Dept. of Computer Science
Iowa State University
Ames, USA
yh54@iastate.edu

Hebi Li
Dept. of Computer Science
Iowa State University
Ames, USA
hebi@iastate.edu

Ge Luo
Dept. of Computer Science
Iowa State University
Ames, USA
gluo@iastate.edu

Forrest Sheng Bao
Dept. of Computer Science
Iowa State University
Ames, USA
forrest.bao@gmail.com

***Abstract*—We propose a pad-focused, net-by-net, two-stage printed circuit board (PCB) routing approach comprising the global routing using Monte Carlo tree search (MCTS) and the detailed routing using A*. Compared with conventional PCB routing algorithms, our approach can route PCB components in both BGA and non-BGA packages. To minimize the gap between the global and detailed routing stages, a polygon-based dynamic routable region partitioning mechanism is introduced. Experimental results show that our approach outperforms state-of-the-art routers such as DeepPCB and FreeRouting in terms of success rate or wirelength.**

***Index Terms*—printed circuit board, global routing, detailed routing, Monte Carlo tree search**

I. Introduction

In designing printed circuit boards (PCBs), routing is a key step placing metal wires for expected connectivity of pins without violating design rules [1]. In recent years, complicated functionality and design goals have sharply increased the amount and density of pins, making PCB routing more challenging. As a result, existing approaches typically break down the problem into two stages: escape routing, focusing on routing in the area around individual components, and area routing, focusing on inter-component routing [2], [3].

However, the two stages do not always couple well. A successful escape routing may make area routing unsolvable [3]. In addition, existing escape and area routing approaches are good at dealing with ball grid array (BGA) chip packages [4] but not non-BGA packages [2], such as passive devices and through-hole pins, which are usually placed irregularly.

Inspired by the global and detailed routing [5] in integrated circuit (IC, chip) design, here we propose a pad-focused, net-by-net, two-stage routing algorithm for PCBs. We partition the routable region based on the locations and dimensions of all pads, regardless of whether they belong to the same components or not, into polygon zones. In the global routing stage, sequences of such polygon zones are assigned to nets while in the detailed routing stage, the polygon assignments are grounded into polylines. By solving pad-to-pad routing directly, our approach optimizes any path regardless of the packaging type of its source or destination. After an assigned polygon is grounded into a path segment in detailed routing, the polygon immediately splits into two by the path to guarantee the equivalence between detailed routability and global routability.

Partially funded by NSF grant CNS-1817089.

Our global and detailed routers are based on Monte Carlo tree search (MCTS) [6] and the A* algorithm [7], respectively. Lastly, to guarantee the connection of all nets, we propose a hierarchical MCTS rip-up-and-reroute mechanism to resolve design rule violations.

II. Method

Our approach (Fig. 1) first routes nets in a random order. For each net, we route two random pins of it first. If the net has more than two pins, we iteratively route the next unrouted pin with a routed pin, randomly drawn, until all pins of the net are visited. After routing a net, detailed routing will change the partitioning for global routing to ensure the equivalency between global routability and detailed routability. This is a major difference between our approach and conventional global-detailed routing split. After all nets are visited, to maximize the connectivity of all nets and satisfy design rules, a hierarchical MCTS-based rip-up-and-reroute mechanism is employed.

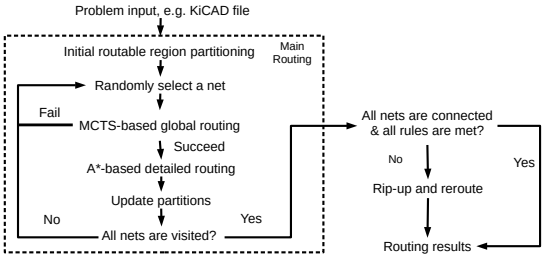

Fig. 1: Overall flow of our approach.

A. Initial routable region partitioning

Each pad or obstacle is approximated as a polygon. By extending all edges of the pad/obstacle polygons to infinity (dash lines in Fig. 2-(a)), the routable region, which is also a polygon, is partitioned into routable polygons. Because the partitioning is performed for each layer independently, the partitions differ from layer to layer.

B. MCTS-based global routing

A global path, a sequence of routable polygons, is constructed from the source polygon to the destination polygon, adding one polygon in the neighborhood of the *head* in each step. Two polygons are neighbors if they share a border on

978-3-9819263-7-8/DATE23/© 2023 EDAA

the same layer or vertically overlap with each other on two consecutive layers. Updated in each step, the head is the latest polygon added to the path. Initially, it is the source polygon. In each step, all the neighbors of the head are evaluated by MCTS [6] and the one with the highest evaluation score is added to the path. If global routing does not find a solution, then our approach will mark the routing of the current net as failed and go to the next net. To quickly check if a global routing solution exists, we use a DFS-backtracking rollout mechanism [8] for MCTS, with which the existence of a global path can be determined after the first iteration of MCTS.

C. A*-based detailed routing

Based on A*, our detailed routing algorithm generates the wires/traces on a grid according to the output of global routing above. Polygons returned by global routing are grounded into paths in the order that they are returned. At the current grid location x, the A* cost function is $f(x) = g(x) + h(x)$ [7]. $g(x)$ can be defined as, for example, $w_1 C_{wl} + w_2 C_{res}$, the weighted sum of the wirelength C_{wl} and the cost of taking routing resources C_{res}. $h(x)$ can be defined as, for example, the Manhattan distance from x to the target. In our experiments, $C_{res} = \max_{p \in \mathcal{P}} \frac{1}{d(x,p)}$ where $\mathcal{P}$ is the set of all the pads/pins, and $d(x, p)$ is the Euclidean distance between x and p. Hence, the grid nodes farther to pads have higher priorities to be routed for the current net. The goal of term C_{res} is to reduce the chance that paths of early-connected nets block the later nets. Other factors, such as the number of vias, can be easily added into $g(x)$ to accommodate different design goals and constraints.

D. Dynamic re-partitioning

Every polygon on a detailed path is split into two by the path (Fig. 2). Repeated at the end of routing a net, this dynamic re-partitioning mechanism ensures a detailed routing solution for each not-yet-routed net if its global routing solution exists.

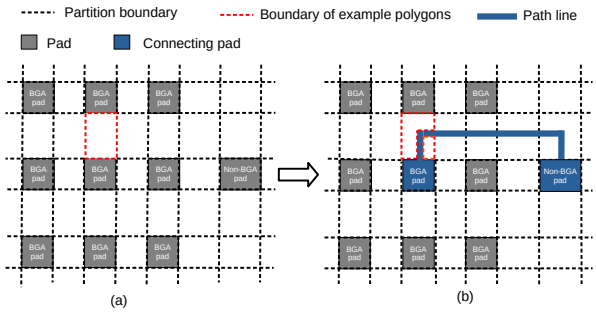

Fig. 2: Routable region partitioning. (a) Initial: edges of pad polygons extended to infinity split routable regions to routable polygons. (b) Re-partitioning after detailed routing: a polygon (red dashes in (a)) assigned to a path (blue, thick) is split into two polygons (red dashes in (b)) by the path.

E. Rip-up and reroute

After all nets are visited, we use a rip-up-and-reroute strategy to reroute pins that are not connected. Given two pins that are not connected as expected, a path connecting them is generated which may violate design rules. Then all paths violating design rules with this new path are ripped and re-routed. The procedure repeats until all the unconnected pins are connected or the maximum number of iterations is reached. Our rip-up-and-reroute is a hierarchical tree search method. Each high-level search tree node represents a global path for a node on a ripped path, and the search method is the depth-first search (DFS) with backtracking [8]. Each global path is searched using the (low-level) MCTS.

III. Experimental Results

Following [2], our approach is compared with two state-of-the-art routers, FreeRouting [9] and DeepPCB [10]. The evaluation was done on 10 real-world, open-source, double-layer PCB designs containing a great mixture of SMD and through-hole pads. Our approach can successfully route all the 10 PCBs, whereas FreeRouting and DeepPCB both fail on two of them. We attribute this to the equivalency between global and detailed routability in our strategy. The baselines underperform our approach in terms of total wirelength but outperform in terms of the number of vias. Lastly, our approach is very extensible to incorporate various constraints and preferences by changing the term C_{res}.

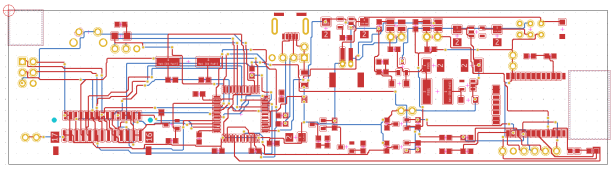

Fig. 3: A PCB routed in our approach.

References

[1] A. B. Kahng, J. Lienig, I. L. Markov, and J. Hu, *VLSI physical design: from graph partitioning to timing closure*. Springer Science & Business Media, 2011.

[2] T.-C. Lin, D. Merrill, Y.-Y. Wu, C. Holtz, and C.-K. Cheng, "A unified printed circuit board routing algorithm with complicated constraints and differential pairs," in *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, ser. ASPDAC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 170–175.

[3] S.-T. Lin, H.-H. Wang, C.-Y. Kuo, Y. Chen, and Y.-L. Li, "A complete pcb routing methodology with concurrent hierarchical routing," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1141–1146.

[4] T. Yan and M. D. F. Wong, "Recent research development in pcb layout," in *2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2010, pp. 398–403.

[5] L.-T. Wang, Y.-W. Chang, and K.-T. T. Cheng, *Electronic design automation: synthesis, verification, and test*. Morgan Kaufmann, 2009.

[6] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, March 2012.

[7] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[8] Y. He, H. Li, T. Jin, and F. S. Bao, "Circuit routing using monte carlo tree search and deep reinforcement learning," in *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 2022, pp. 1–5.

[9] "Freerouting," https://freerouting.org/.

[10] "DeepPCB," https://www.deeppcb.ai/.