A Global Router with Topology Optimization on Hanan Grid

Rui Wang
School of Information Engineering
Wuhan University of Technology
Wuhan, China
rui.wang@whut.edu.cn

Ning Xu
School of Information Engineering
Wuhan University of Technology
Wuhan, China
xuning@whut.edu.cn

Jixin Zhang*
School of Computer Science
Hubei University of Technology
Wuhan, China
zhangjx@hbut.edu.cn
*Corresponding author

2024 13th International Conference on Communications, Circuits and Systems (ICCCAS) | 979-8-3503-8627-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCCAS62034.2024.10652868

***Abstract*—Global routing not only becomes an essential part in the domain of EDA backend routing but also serves as a method to quickly assess routing resources. Also, it dominates propelling and guiding detailed routing. Unluckily, minimization of overflow on PCB boards remains more challenging for a great majority of global routing algorithms due to the flexibility of PCB routing and the complexity of net topology. A novel global routing approach is put forward here at this point. Its process involves establishing a Hanan grid to represent spatial resources, applying the A* algorithm with rip-up and reroute for congestion negotiation. Meanwhile, a topology optimization model is introduced to explore more optimal topologies for minimization of overflow. Our measurements indicate that our global routing approach can achieve higher routing rate in less time in comparison with other global routing methods.**

Keywords—Printed circuit board, Global routing, Hanan grid, Topology optimization

I. INTRODUCTION

Routing that is a complex and time-consuming task is priority number one for physical design of PCBs. Its processes are made up of global routing and detailed routing.

As for the former, nets are connected on a coarse-grained grid map with capacity constraints and routing solutions are provided for the latter. It is a crucial stage in PCB design because it greatly affects time and quality of the latter. In order to reduce power dissipation and crosstalk delay in the circuit, in addition to reasonable bus encoding schemes [1], the global routing solution will also have an impact. It may even affect the speed and number of iterations in the design cycle, so that it is a crucial requisite to determine circuit performances [2].

Steiner trees or minimum spanning trees [3] are typically employed for global routing. Lin S-ED and Kim DH [4] aimed to construct all Steiner trees on Hanan grids in order to carry out selections based on various congestion situations. Monzurul Islam Dewan et al. [5] proposed topology database size reduction techniques for practical application of the database [4]. The Steiner tree is one of the most popular routing models while Steiner minimum trees are frequently used for creation of the initial topology of non-critical networks. Han et al. studied the capacity minimum spanning tree problem to achieve global optimization [6]. Regrettably, the input volume of the Steiner tree or minimum spanning tree problem would increase along with growth of the size of the routing problem.

With the advancement of circuit technology, there are more and more pins and multi-terminal nets in circuits, so that there may be more limitations in traditional routing or minimum Steiner tree algorithms. Thus, these algorithms are not fit for this type of multi-objective mission.

Luckily, the global routing optimization algorithms are becoming more popular so far, which are categorized into two primary type of the multi-channel flow-based technology and the rip-up and reroute technology.

As for the former, the problem of global routing is described as a multi-commodity flow problem so that it can be equivalently regarded as an integer linear programming (ILP) problem. Tiago Augusto Fontana et al. [7] applied their ILP-based technology to optimize global routing based on simultaneous movement of cells and routes nets. BoxRouter [8] extended a box that initially covers the most congested area and routed all nets within the box through by means of ILP. Multi-channel flows are challenging due to its great growth of time complexity.

By comparison, the latter are more popular for optimization of paths. The maze algorithm can be applied to find the shortest path connecting two pins in the presence of obstacles. The A* algorithm that is a heuristic search algorithm can be applied to not only find the shortest path but also efficiently reduce the search time. Rip-up and reroute involves sequentially connecting each net without taking congestion into consideration initially. After establishing connectivity among all nets, it is imperative to identify the congested areas. All nets in these areas should be rip-up and rerouted to eliminate congestion. Additionally, it can be used in combination with the global routing method as a post-processing step to enhance routing quality.

The effectiveness of 2 algorithms (namely DpRouter [9] and FastRoute [10]) was demonstrated. J. Liu et al. [11] employed the Steiner tree to perform a rough assessment of resources, followed by application of the maze algorithm to research the path. This approach enables a sequential more precise evaluation of resources, resulting in a rapid and effective evaluation process.

SpRouter [12] utilized the pattern routing as the initial route, leveraging maze routing and rip-up and reroute techniques to generate improved paths for each net. P. Yao et al. [13] applied ILP to develop a path-finding model and employed the A* algorithm along with rip-up and reroute techniques to get the routing results. Their findings demonstrate that overflow can be reduced significantly by means of such approach and routing results can be improved.

During the global routing process, Hanan grid is becoming more popular, where those methods such as Steiner trees are applied. Pei-Ci Wu et al. [14] put forward a method for dynamically decomposing large grids to achieve multiple uses of grid edges for path finding. However, their method is time-consuming and requires continuous decomposition of the grid during the routing process.

Yan J-T [15] implemented his grid approach, where the grid is regarded as nodes of path-searching map and a directed graph structure based on Depth-First Search (DFS) is employed so as to effectively solve the cross problem of paths. Additionally, it incorporates a capacity control function to further enhance its performance. Thankfully, the issue of bus routing was solved specifically by mean of such method. It has been determined that there shall be no cross between various buses. Namely, it would not provide a solution for the extensive routing of non-bus nets.

Edge nodes are utilized as routable nodes in a traditional Hanan grid. Qinghai Liu et al. [16] used the grid center nodes to control the path angle. Routing quality can be improved by means of the maze path search algorithm which can help to improve the routing congestion adjustment. Zou et al. [17] applied a queue-based data structure to implement congestion-aware maze routing, allowing a node to be visited more than once. Thus, routing can be speeded up.

In addition, topological cross is an important factor that affects the routing results. If the topologies of various networks had been crossing, the non-cross-connection would never be completed. Wu et al. [18] determined whether the topology was crossed by removal of the same symbols after sequence expansion. However, it is difficult to determine the topology of all networks on the entire board, and a method is needed to generate the non-cross topology of the entire board.

Most global routing techniques focus on routing the bus for PCB boards. There is no unified global routing algorithm for the entire PCB board up to this point. When global routing is performed on the PCB board, pattern routing cannot give rise to a small overflow so that guidance of detailed routing can be little affected. Moreover, the growth of the number of pins in modern PCB multi-terminal nets may result in more complex topologies. It is difficult to find a good topology based on Steiner trees.

A fast, flexible, and high routing rate algorithm was put forward here. First, the A*algorithm is applied to find the path and a graph is produced to serve as the A* map for the Hanan grid. No dynamical grid decomposition occurs. Thus, there would be fewer path points than those of the uniform grid to not only ensure a high routing rate but also accelerate path finding. The resulting congestion and cross can be solved. Overflow can be solved by means of additional rip-up and reroute methods. The A* algorithm is employed throughout the entirety of our global routing process where the Hanan grid is regarded as the platform for evaluating resources. Our method not only ensures consistency in the cost calculation process throughout the global routing process but also takes advantages of greater flexibility and saving time in scenarios such as global routing, where strict line length requirements are not imposed. Moreover, it is important to note that a single topology may not be able to optimize results with minimal overflow, especially in cases where there are topological cross. A topology optimization model is established to predict whether a new topology can achieve a good routing rate. By changing the topology and choosing the most appropriate one, global routing can be optimized.

The main contributions of our algorithm are as follows:

1. Creation of a graph for the Hanan grid, and application of the A* algorithm to find the path on the graph.

2. Identification of congestion and cross on the graph, and achievement of rip-up and reroute to reduce overflow.

3. Establishment of a topology optimization model and selection of an improved topology to achieve better routing results.

II. FRAMEWORK

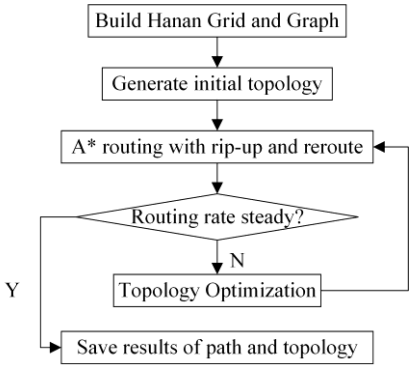


Fig. 1. Overall framework.

Our global router is schematically shown in Fig. 1. First, a Hanan grid is built, where its edges are regarded as the graph's nodes and the physical distance between the edges are utilized as the weight, respectively. An initial topology is then generated for each net based on the input net list sequence. A* uses the graph as its map, and the initial topology is applied for routing. Rip-up and reroute techniques are employed to resolve congestion. If the routing rate remains unchanged, topology optimization shall be performed and alternative topologies shall be selected to perform trial once more to optimize our results. While the routing rate of the topology remains unchanging after performance of multiple changes, it shall be terminated.

Algorithm 1 outputs the pseudo-code of our global routing algorithm.

Algorithm 1 Our global routing algorithm

Input: routing graph G, all net topologies
Output: global routing solution for all nets
1 while routing rate are not steady
2 choose an optimal topology;
3 for layer from all layers, do

```
4                   find paths by using A* rip-up and reroute on the layer;
5                   if routing rate = 100%, then
6                           return global routing solution;
7                   end
8                   input nets with overflow to the next layer;
9           end
10 end
11 return global routing solution
```

III. Our Technique

A. Map of Global Routing

The Hanan grid refers to grids that are created by extending and intersecting 4 sides of the external rectangle encompassing all modules. For minimization of the quantity of grids and removal of numerous small grids, the extension lines will be obstructed by other modules. In PCB, a pin is defined as a module. The grid is shown in Fig. 2.

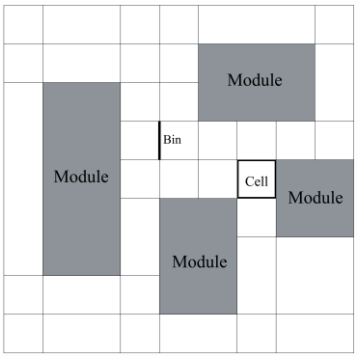
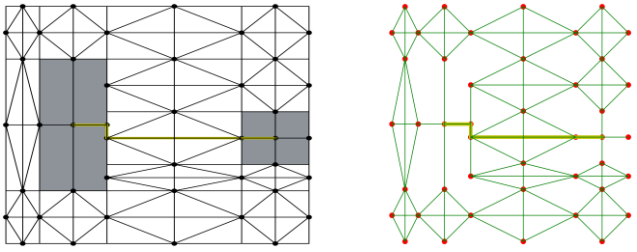


Fig. 2. Hanan gird and example of “Bin” and “Cell”

After our Hanan grid has been constructed, each grid is referred to as a "Cell". Each edge of the "Cell" is recorded as a "Bin", which will be used on the path finding map for A*. A graph is constructed on the Hanan grid to serve as a map for path searching. The graph's vertices represent the "Bins" of the Hanan grid, while the edges represent the connections between these "Bins". The weight of each edge corresponds to the distance between 2 adjacent "Bins". Additionally, the pin and the midpoints around it will also serve as vertices to add the pin to the graph and establish the edges. The A* algorithm will start from the starting "Bin", and based on the established edges and costs. Then, the lowest cost path is found near the ending "Bin" in the graph. As shown in Fig. 3, only a few vertices are necessary to connect a pair of pins.



(a) (b)

Fig. 3. (a) The Graph on the Hanan grid and one path. (b)Vertices, edges and one path on the graph.

Because a "Bin" can accommodate multiple paths, the A* algorithm is capable of searching for "Bins" with existing paths during our path finding period. The corresponding problem occurs, it is necessary to determine the number of paths where a "Bin" can accommodate to assess congestion, as well as the cross of paths within the "Cell".

(1) Capacity and Congestion State of "Bin": The length of the "Bin" is referred to as the "maximum capacity". In case of routing, the capacity of the "Bin" occupied by each path is determined by the sum of the path width and spacing. The remaining available capacity of the "Bin" can directly reflect the level of congestion in the area around the "Bin". A trial is performed to find a "Bin" that can accommodate the required capacity for the path. If the remaining capacity of the "Bin" is less than zero, the "Bin" is defined as being in a "congested state".

Through the above method, it can be directly determined whether there is congestion during the routing process.

(2) Cross of "Bin": Unlike path finding methods such as uniform lattice map and line exploration, the grid map remains challenging in determining cross within the grid. A non-uniform grid is centered, where each "Bin" has its capacity. This allows various paths to traverse the same "Bin", but the incoming and outgoing lines may be in various directions, potentially leading to cross. There are three cross situations (Fig. 4).

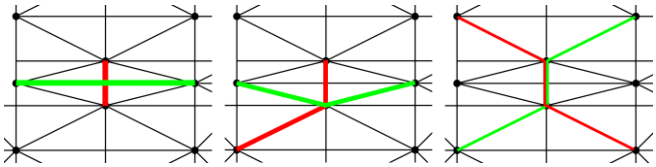


Fig. 4. Three cross situations

Various scenarios of cross and their corresponding processing methods are listed as follows:

① Intersection of horizontal and vertical wires within a "Cell": It is important to note that horizontal and vertical paths cannot coexist within a grid by default.

② Crossing of 2 sections of routing inside the "Cell": By default, a path is not allowed to find paths twice in a "Cell" at the same time.

③ "co-Bins" cross, that is, 2 paths of nets have met on the same "Bin", and the direction of entering the "Bin" and of exiting the "Bin" are cross: It is determined whether there is cross by calculating the direction difference of 2 paths of nets entering and exiting the same "Bin". The determination process of "co-Bins" cross is shown in Fig. 5.

Fig. 6 shows examples of a case including direction difference. In Fig. 6 (a), Paths 1 and 2 do not cross, while 2 paths do cross in Fig. 6 (b). It can be seen that when D1 and D2 remain the same, cross does not occur. On the other hand, if they are different, cross will happen.

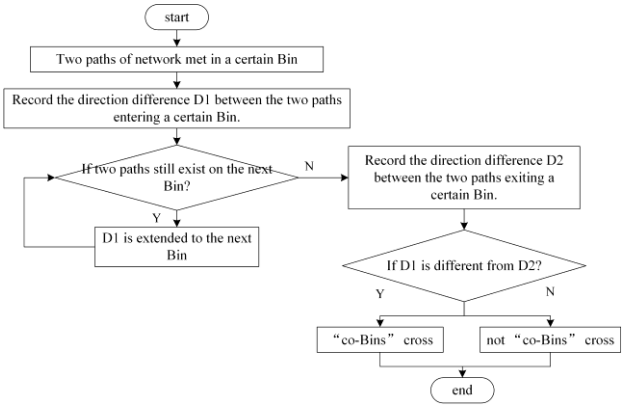


Fig. 5. Flow diagram of finding "co-Bins" cross

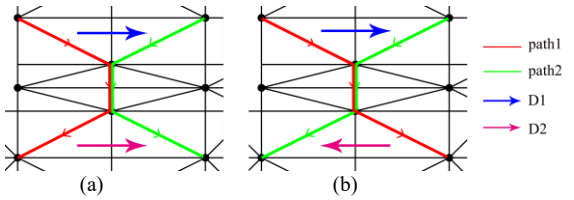


Fig. 6. Examples of direction difference D1 and D2

B. Routing

Our A* algorithm is applied for conduction of our path finding, where the graph is utilized as the map. Our routing is schematically shown in Fig. 7. Our rip-up and reroute is applied to relieve congestion. During the initial round of our routing, overflow is allowable. The subsequent routing processes will involve calculation of congestion and cross costs. Since effects of a single "Bin" crossing on the route rate is worse than that of multiple "Bin" congestion, it is logical to assign a higher cost to the cross rather than the congestion. Simultaneously, variations in the distance between any 2 "Bins" are influenced by various "Cell" sizes, resulting in discrepancies in the dimensions of distance and congestion costs. Thus, it is necessary to consider effects of the "Cell" size on congestion and cross costs so that a tradeoff can be achieved between the various dimensions of costs.

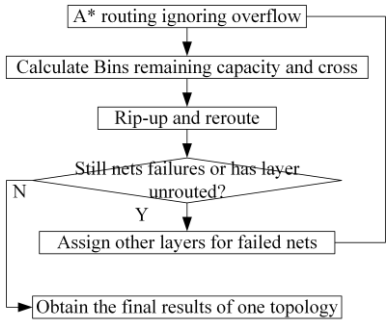


Fig. 7. Flow diagram of routing

The calculation methods for assessing congestion and cross costs for each "Bin" are as follows:

$$Congest\,\mathrm{Cost} = \lambda * CellScale * Capacity\,\mathrm{Cost} \quad (1)$$

$$Cross\,\mathrm{Cost} = n * \lambda * CellScale * CrossPenalty \quad (2)$$

where: λ represents a positive integer greater than 0, while n represents a positive integer greater than 1. The specific values of λ and n are determined based on the specific circumstances. "Cell Scale" is the scale of the grid where "Bin" is located, which is expressed by the longest side length of "Cell".

$$Capacity\,\mathrm{Cost} = \frac{|RCapacity - WCapacity|}{WCapacity} \quad (3)$$

where: RCapacity is "Remaining Capacity" of one "Bin". WCapacity is "Wire Capacity" of one path. "Remaining Capacity" represents the amount of available capacity that remains in the "Bin" while "Wire Capacity" denotes the capacity required for a single path. "Capacity Cost" represents the cost associated with the capacity of the "Bin", whose calculation is determined by the above equation in case that the situation of "Remaining Capacity" is less than "Wire Capacity".

If the remaining available capacity is insufficient to accommodate one additional path, it indicates that the "Bin" is unable to accommodate any more paths. The "Capacity Cost" variable represents the number of affected nets.

"Cross Penalty" denotes the number of nets that are impacted by the cross on the "Bin".

After each net is routed, the net flag of the path is appended to the "Bins" that the path traverses. The remaining capacity of the "Bins" is reduced by the spacing required for this path. After all paths are laid out, it can be calculated whether the remaining capacity of all "Bin" of paths is negative. If a negative value is encountered, the "Bin" should be recorded and a "History Cost" which should be added. If the "Bin" is less than 0 in the subsequent round, it will continue to experience congestion or cross. Consequently, the historical costs will rise. During the subsequent phase of routing, all paths belonging to the same net shall be removed and they shall be rerouted. Through a series of iterations, the paths on the map are continuously adjusted to minimize congestion and cross, until all congested and cross "Bins" are removed.

The cost equation of the A* algorithm is f(n) = g(n) + h(n), where g(n) represents the cost of the path from the starting point to the current point, while h(n) represents the cost of the path from the current point to the end point. The function g(n) is computed as follows:

$$g(n) = Dis\tan ce\,\mathrm{Cost} + Congest\,\mathrm{Cost} + Cross\,\mathrm{Cost} + History\,\mathrm{Cost} \quad (4)$$

Layer properties for "Bin" are added. Although the layout consists of multiple layers, the process of path finding involves routing on a single layer of the graph. Since the number of "Bins" remains unchanged, a large number of layer-changing search processes can be reduced and route time can be saved. If it is not possible to route all nets on a single layer during the routing process, the nets that are causing congestion and cross will be routed on various layers until all layers have been successfully routed. If there are any remaining nets that cannot be deployed, it is advisable to consider modification of the topology.

C. *Topology optimization*

When a net possesses three or more pins, various topological connection methods can be employed, such as Y-shaped or Star-shaped configurations. In case that there is limited availability of space resources and the routing rate is low, it is often advantageous to modify the net topology to get improved results.

There are several characteristics associated with a topology, such as the length of the topological flying line, the placement of the module, and the presence of nearest/farthest pin connections within the net. Various characteristics may affect the routing rate to varying degrees. It is challenging to accurately predict the correlation between topology and routing rate based on a single characteristic. If multiple features represent topology together, the correlation with routing rate can be significantly improved. If the features are quantified and their correlation where the routing rate is determined as a weight, the weighted sum of these features can be utilized to predict the association between a topology and the routing rate. For instance, if a feature has a positive weight that a large value, the higher the value of the feature in a specific topology is, the more probable its higher routing rate is.

Linear regression is employed and a large number of open source cases and cases from industry are verified, Each case includes a large number of characters of different topologies and forms a training set with the corresponding distribution routing rate. Our linear regression model is trained to obtain weights of various characters, which form the correlation with routing rate.

After this model has been applied and a new topology has been randomly generated, corresponding prediction of the routing rate of the new topology is performed based on the model. If it has a higher predicted routing rate, this topology is applied to test our routing. If the predicted routing rate decreases, it will be randomized again. For removal of time wastage resulting from inefficient topology routing, it is imperative to generate novel topologies.

IV. Experiment Results

Our global router is implemented in C++, the experiments are performed on a 64-bit Windows workstation equipped with AMD Ryzen 7 5800H 8-CoreProcessor 3.2 GHz, and 32 GB RAM. Open source PCB benchmark [19] was tested to examine effects of these cases on our global router. Each case has a different number of nets and layout sizes (Table I).

TABLE I. Scale of Cases

Cases	Area	Layer	Pins	Nets
bm1	1016*533	2	319	99
bm2	508*228	2	75	34
bm3	39 *350	2	229	80
bm4	400*410	2	161	54
bm6	220*600	2	140	52
bm7	205*139	2	40	15
bm8	564*863	2	188	70
bm9	860*715	4	312	63
bm10	580*595	4	233	35

Comparisons were carried out between other global routers, which are the DpRouter (pattern routing). BoxRouter 2.0 (Steiner trees coupled with A* and rip-up and reroute technique) and dynamic meshing method [14] coupled with A*. Various factors such as time (T), wire length (WL), and routing rate (RR) which represents the proportion of the nets that is fully connected were measured so that no overflow shall occur.

The corresponding results are shown in Table II. Results indicate that DpRouter can output results quickly due to its pattern routing. Unluckily, it is difficult to remove overflow results on the PCB board due to its fixed linear direction so that it shall be difficult to maintain a high routing rate. BoxRouter 2.0 can maintain a higher routing rate in a shorter period. But it is difficult to choose a superior Steiner tree for nets made up of dozens of pins, which makes it challenging to perform optimization and maintain a higher routing rate. Dynamic meshing requires a certain amount of time to continuously divide the mesh, and there may be smaller grids that are lack of sufficient capacity to accommodate a path. Consequently, the space resources in these grids remain idle, leading to a wastage of space resources and a decrease in the routing rate.

Table III presents a comparison between our global router in case of application of a single topology and topology optimization. The recorded results are the averages when the optimal routing rate is first reached. Fig. 8 shows the results of our global router (bm7).

TABLE III. Results of Our Global Router

Cases	Our global router +topology optimization			
	Single Topology RR	Ave T (s)	Ave WL	Ave RR
bm1	0.91	168.9	77134	0.95
bm2	1	1.4	3,886	1
bm3	0.93	50.9	20,919	0.97
bm4	0.92	68.1	17249	0.97
bm6	0.95	28.9	12118	0.98
bm7	1	0.1	1138	1
bm8	1	5.8	17236	1
bm9	0.98	103.8	89082	0.99
bm10	0.99	52.9	35196	1

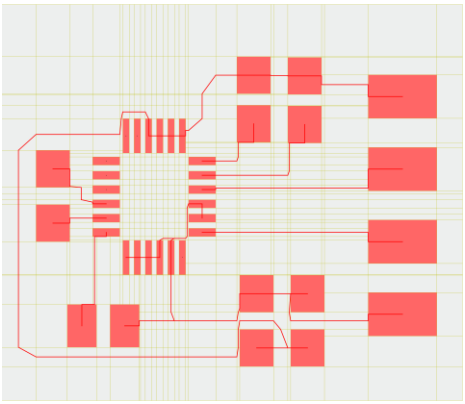


Fig. 8. bm7 global routing result.

TABLE II. RESULTS OF SOME GLOBAL ROUTING METHODS

Cases	DpRouter			BoxRouter 2.0			Dynamic Meshing [14] + A*		
	T (s)	WL	RR	T (s)	WL	RR	T (s)	WL	RR
bm1	2.01	113928	0.62	49.6	112302	0.92	367	100645	0.66
bm2	0.03	6566	0.83	1.2	6136	0.97	19	4888	0.97
bm3	0.02	37988	0.72	24.1	35858	0.92	23	24820	0.80
bm4	0.59	37953	0.65	41.8	28985	0.93	23.2	21007	0.74
bm6	0.46	19583	0.69	15.8	18557	0.92	16.4	15371	0.82
bm7	0.01	1987	1	0.2	1687	1	0.6	1249	1
bm8	0.61	35874	0.85	3.2	32419	0.94	55	21588	0.97
bm9	3.57	153468	0.73	51.5	143578	0.88	1106	145719	0.86
bm10	1.19	68725	0.71	32.7	52894	0.89	257	59407	0.90

It can be observed that a high routing rate cannot be easily maintained by means of a single topology in case of dealing with a large number of nets or limited resources. After the topology optimization has been applied, some improved topologies can be discovered, which possibly increases the routing rate at the expense of growing time. Considering the fact that our method requires only a minimal amount of time per round of routing, this slight increase in time is allowed. It is evident that the implementation of more efficient topologies in our global router can lead to a remarkable improvement in routing rate, less wire length or a shorter time.

V. CONCLUSION

A novel global router is presented here, where the Hanan grid is implemented for the PCB layout and the grid edges are regarded as nodes of map. The A* algorithm is applied for path finding to successfully and quickly accomplish the global routing of the entire PCB board. Based on extraction of topological characters and training our topology optimization model, it is possible to proactively select some more efficient topologies to obtain results in case of higher routing rates. Thereby, ineffective attempts of routing can be avoided. The open source PCB benchmark is utilized for measurements. Compared with the conventional global routers, our method enables a higher routing rate of PCB within a short time.

The routing speed is influenced by the number of Hanan grids, so a poor layout will impact the global router's efficiency. Our future work will focus on reducing the number of grids to improve routing efficiency, potentially in combination with placement algorithms.

REFERENCES

[1] K Padmapriya, "MODIFIED BUS INVERT TECHNIQUE FOR LOW POWER VLSI DESIGN IN DSM TECHNOLOGY," International Journal of Electrical and Electronic Engineering & Telecommunications, Vol. 2, No. 2, pp. 54-57,April 2013.

[2] H. Tang, G. Liu, X. Chen and N. Xiong, "A Survey on Steiner Tree Construction and Global Routing for VLSI Design," in IEEE Access, vol. 8, pp. 68593-68622, 2020.

[3] P. P. Guha Neogi, "Novel Approach for Steiner Nodes Filtration in VLSI Global Routing," 2019 International Conference on Intelligent Computing and Remote Sensing (ICICRS), Bhubaneswar, India, pp. 1-6, 2019.

[4] Sheng-En David Lin and Dae Hyun Kim. Construction of All Multilayer Monolithic Rectilinear Steiner Minimum Trees on the 3D Hanan Grid for Monolithic 3D IC Routing. In Proceedings of the 2019 International Symposium on Physical Design (ISPD '19). Association for Computing Machinery, New York, NY, USA, 57–64, 2019.

[5] Monzurul Islam Dewan, Sheng-En David Lin, and Dae Hyun Kim. Construction of All Multilayer Monolithic RSMTs and Its Application to Monolithic 3D IC Routing. ACM Trans. Des. Autom. Electron. Syst. Just Accepted (October 2023), 2023.

[6] HAN Jun, FANG Qingan, MAO Liyong, et al., "A Hybrid Optimization Algorithm for the CMST Problem," Chinese Journal of Electronics, vol. 20, no. 4, pp. 583-589, 2011.

[7] T. A. Fontana et al., "ILPGRC: ILP-Based Global Routing Optimization With Cell Movements," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2023.

[8] Minsik Cho, Katrina Lu, Kun Yuan, and David Z. Pan. BoxRouter 2.0: A hybrid and robust global router with layer assignment for routability. ACM Trans. Des. Autom. Electron. Syst. 14, 2, Article 32 (March 2009), 21 pages, 2009.

[9] Z. Cao, T. Jing, J. Xiong, Y. Hu, L. He and X. Hong, "DpRouter: A Fast and Accurate Dynamic-Pattern-Based Global Routing Algorithm," 2007 Asia and South Pacific Design Automation Conference, Yokohama, Japan, pp. 256-261, 2007.

[10] Yue Xu, Yanheng Zhang and Chris Chu, "FastRoute 4.0: Global router with efficient via minimization," 2009 Asia and South Pacific Design Automation Conference, Yokohama, pp. 576-581, 2009.

[11] J. Liu, C. -W. Pui, F. Wang and E. F. Y. Young, "CUGR: Detailed-Routability-Driven 3D Global Routing with Probabilistic Resource Model," 2020 57th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA,pp. 1-6, 2020.

[12] J. He, U. Agarwal, Y. Yang, R. Manohar and K. Pingali, "SPRoute 2.0: A detailed-routability-driven deterministic parallel global router with soft capacity," 2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC), Taipei, Taiwan, pp. 586-591, 2022.

[13] P. Yao, P. Zhang and W. Zhu, "Pathfinding Model and Lagrangian-Based Global Routing," 2023 60th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, pp. 1-6, 2023.

[14] Pei-Ci Wu, Q. Ma and M. D. F. Wong, "An ILP-based automatic bus planner for dense PCBs," 2013 18th Asia and South Pacific Design Automation Conference (ASP-DAC), Yokohama, Japan, pp. 181-186, 2013.

[15] J. -T. Yan, "Single-Layer Obstacle-Aware Multiple-Bus Routing Considering Simultaneous Escape Length," in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 12, no. 6, pp. 902-915, 2022

[16] Q. Liu et al., "Disjoint-Path and Golden-Pin Based Irregular PCB Routing with Complex Constraints," 2023 60th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, pp. 1-6, 2023.

[17] P. Zou, Z. Cai, Z. Lin, C. Ma, J. Yu and J. Chen, "Incremental 3-D Global Routing Considering Cell Movement and Complex Routing Constraints," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 42, no. 6, pp. 2016-2029, 2023.

[18] WU Haoying, ZOU Sizhan, XU Ning, et al., "A Bus Planning Algorithm for FPC Design in Complex Scenarios," Chinese Journal of Electronics, in press, 2024.

[19] PCBBenchmarks.2020. https://github.com/aspdac-submission-pcblayout/PCBBenchmark