On Routing Fixed Escaped Boundary Pins for High Speed Boards

Tsung-Ying Tsai
Global UniChip Corp.
Hsinchu, Taiwan

Ren-Jie Lee, Ching-Yu Chin
Chung-Yi Kuan, Hung-Ming Chen
Dept. of Electronics Engr.
NCTU, Hsinchu, Taiwan

Yoji Kajitani
Faculty of Environmental Engineering
The University of Kitakyushu
Kitakyushu, Japan

Abstract— Routing for high speed boards is still achieved manually nowadays. There have been some related works in escape routing to solve this problem recently, however a more practical problem is not addressed. Usually the packages/components are designed with or without the requirement from board designers, and the boundary pins are usually fixed or advised to follow when the board design starts. Previous works in escape routing are not likely to be used due to this nature, in this work, we describe this fixed ordering boundary pin escaping problem, and propose a practical approach to solve it. Not only can we have a way to address, we also further plan the wires in a better way to preserve the precious routing resources in the limited number of layers on the board, and to effectively deal with obstacles. our approach has different feature compared with conventional shortest-path-based routing paradigm. In addition, we consider length-matching requirement and wire shape resemblance for high speed signal routes on board. Our results show that we can utilize routing resource very carefully, and can account for the resemblance of nets in the presence of the obstacles. Our approach is workable for board busses as well.

I. Introduction

Today we have increasing pin count on the very dense boards, and current CAD tools for board routing are very few and can not provide automatic solutions on board-level routing. There have been quite a few works addressing this problem, such as [4], [5], [9], [11], [12], [13], [14], [15], [16], [17], [8], [10]. The key problem in board-level routing is escape routing, which includes the escape from inner pins to the boundary and from the boundary to other components.

The first escape routing problem is to route the wires from the pins located close the center of the component to escape to the slots which are terminals on the component boundary [11], [14], [15], [8], among which [8] proposes a problem that has a constraint on pin ordering from the requirement of the board designers. The second escape routing problem (sometimes is referred to area routing) is to connect the escaped slots between components/packages on the board [4], [13], [16]. In order to have better solutions, very recent works combine these two problems to have larger solution space to explore, called "simultaneous escape routing" [9], [10]. [10] uses negotiated congestion based router to achieve the routing, this technique is widely used in modern academic global routers. On the other hand, [9] is featuring a new concept "boundary routing" to solve simultaneous escape routing problem.

In this paper, we want to address a different yet practical problem: how we route the wires or busses on the board with the fixed components/packages boundary pins. Usually the packages/components are designed separately with boards, even with few interactions in codesign, they are shipped with fixed boundary pins. Another occasion is that those close-to-center pins in the component are pre-escaped ([18], [17]) by the provider, and board designers are advised to use the given order of the escaped boundary pins. It is worth mentioning that the work in [8] can not really resolve the problem here since its mission is to solve the given order of the boundary pins for board area routing, which is on the opposite of the direction of wire planning.

Our work is a two-stage planner: the first stage is to try to obtain a planar topology for all the connections with more routing space available in subsequent pin pair routing; the second stage is to obtain a refined routing with length-matching constraint awareness and wire shape resemblance. In addition, our approach is workable for board busses as well, even with routing obstacles. The remainder of this work is organized as follows. Seciton II describes the board routing issues. Section III describes the first stage of our routing methodology to obtain planar topology. Section IV illustrates the routing refinement for length constraints and wire shape resemblance. Section V shows our results followed by the conclusion in Section VI.

II. Routing Issues for High Speed Boards

Manual work is adpoted in board-level routing due to many issues and constraints. The first concerning issue is impedance. Considering the IR drop, crosstalk and impedance matching, the net shape and the net length must be controlled based on certain criteria. Many original ideas have been created to solve these problems, such as [6], [12], [13], [16]. BSG-router [16] proposed an effective routing method. This method can match the net length and save a large amount of memory for data under given routing topology. However it consumes enormous routing area and ignores the shape of nets, creating a lot of jogged nets (net-jogging). These net-joggings may cause signal integrity issues.

The second problem is bus planning. Bus router handles a set of nets and routes these nets in parallel. An automatic bus planner is proposed in [4], which provides good results in bus planning. It is based on a good escaped pin assignment, therefore the bus planning can be automated. But from the observation of the results, the buses are planned in a bunch, it seems unfriendly to obstacles and routing resources.

Therefore the consequent problem is routing resource in board routing. Due to manufacturing cost, we should keep the number of layer used minimum, which generates the constraint of planar routing for board-level planning. Most of the routing algorithms are based on the concept of shortest path. This kind of router cannot take the whole set of nets into account, in each step only the present net will be considered and routed. Since the routing of present net may block the routing path (splitting the routing space) in sequential routing framework and planar requirement, the routing of the following net or other nets is not favored due to early routing decision. Since the shortest path algorithm usually causes the failure in such sequential routing, we are forced to go back to the previous step and re-route the net. [2] provides a good framework for topological routing, however it must

This work was partially supported by the National Science Council of Taiwan ROC under grants No. NSC 97-2221-E-009-174-MY3 and 99-2220-E-009-019.

978-3-9810801-7-9/DATE11/(c)2011 EDAA

use partitioning and Delaunay triangulation for application, and it also has net ordering problem. In very recent work [9], boundary routing is a very good concept, but it may be limited by only several escape patterns.

Considering the fixed boundary pin planar routing, our approach will try to mitigate the aforementioned effects. In order not to spoil the routing space for the rest of the nets and to avoid the net crossing during board wire planning, our first stage routing will optimize the routing order to eliminate the conflicts of net routing and to better utilize the routing resource, which is very different from traditional IC global router. During the second stage refinement, we propose length-constraint-aware heuristics to be able to achieve min-max length bounds of nets, and take care of wire shape resemblance.

III. Routing under Fixed-Ordering Pin Locations

In this section, we propose our stage of routing under fixed-ordering boundary pins. First, we decide the net ordering statically by dynamic pin sequence (DPS) to avoid the net crossing in one-layer planar routing. Second, we perform the topological routing under the aforementioned net order. We develop an against-the-wall routing for better preservation of routing resource for consequent routings [1]. Last, we describe how we handle the inevitable crossing nets.

A. Static Net Ordering with Dynamic Pin Sequence

Before we present the routing algorithms, we first describe the approach for net ordering determination. Given the placement of the components on the board, routing space is formed with boundary pins on the periphery of the components. The pins are either fixed by package providers or escaped using methods such as [18] [17]. We use a number sequence to represent the relative position of pins on one component. In order to include these pins in a number sequence, we randomly select a pin among the pins on one component as the start point of the sequence. From this start point, other pins of this component are inserted into the sequence one by one in counterclockwise fashion. We apply the same process to the other components and get a series of pin-sequence. As Fig. 1 shows, this component contains five pins, we choose the pin A as the start point and make a pin-sequence in counterclockwise fashion, the sequence is $ABCDE$.

With components transforming to pin-sequences, we will concatenate those sequences into one sequence, we name it dynamic pin sequence (DPS). The idea is to decide one pin-sequence among all pin-sequences as a basis sequence, then we put other pin-sequences into this basis by finding a position in the sequence, we refer it as component connecting point (CCP). As illustrated in Fig. 2, pin K is treated as a CCP, and we can concatenate two sequences as one. In this example, if the second (on the bottom of Fig. 2) component's pin-sequence is $NABCDEFGHIJKLM$ and the first component's pin-sequence is $ANMLKJIHGFEDCB$, the concatenated sequence is $NABCDEFGHIJKKJIHGFEDCBANMLLM$ when choosing the former sequence as a basis sequence (base DPS). There are many ways/combinations for sequence concatenation, we choose a greedy way. By applying the longest pin-sequence (the component with the largest number of pins) as a basis, we then integrate the second-longest pin-sequence into the basis, and then the third-longest one, and so forth[2].

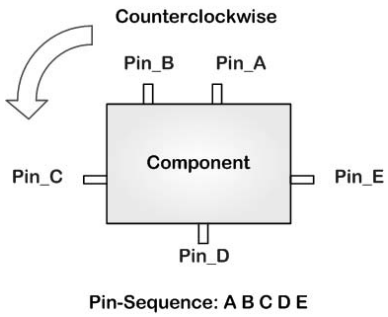

Fig. 1. An illustration of pin-sequence generation of a component for PCB routing. Pin A is chosen as the first pin. By walking along the component boundary counterclockwise, the other pins are listed as the pin-sequence in order.

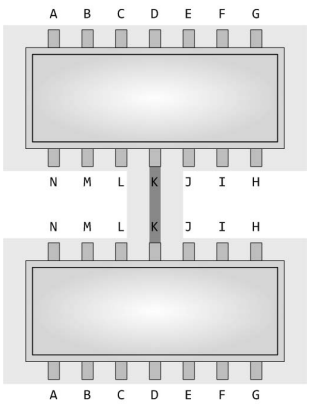

Fig. 2. An illustration of a component connecting point (CCP), which can be seen as the first connection among all nets in components, is chosen from the pin-sequences of the components. Pin K is set to be the CCP and connected to the corresponding component.

Two consequent problems remains: how do we use this concatenated DPS for net ordering, and which position/pin location is the best? We will first describe how we process the DPS for the first question, then we will present how we choose the CCP for sequences concatenation. Using Fig. 3 as an example, we can see that the pin-sequence C_1 is put into pin-sequence C_2 (C_2 is the base DPS in this case). Pin B is chosen as the CCP for the concatenation of two sequences (Fig 3(b)), the resultant sequence C_2 is $ABBACCHIGEFD$ (Fig. 3(c)). In Fig. 4, the final DPS is $ABBACCHIGGIHEFDDEF$. We observe that symbol B is repeated due to the concatenation, this means we can have the routing connection of two pins in the netlist. We then "erase" these two symbols in the sequence and the sequence becomes $AACC$.... It is obvious that there are two sets of repeated symbols: $AACC$, this means net A and net C can be connected consequently without crossing. Same principles

[1] B-escape [9] is a very recent work with very good results in solving the simultaneous pin assignment and routing problem. However we make the following observations. First, our concept of against-the-wall routing is somewhat similar to the boundary routing but different. Second, our approach has no limit on monotonic routing, and has few detours.

[2] It is possible to have more than one DPS since there are many components which connect to different components. Larger components (containing more pins) get to form other DPSs.

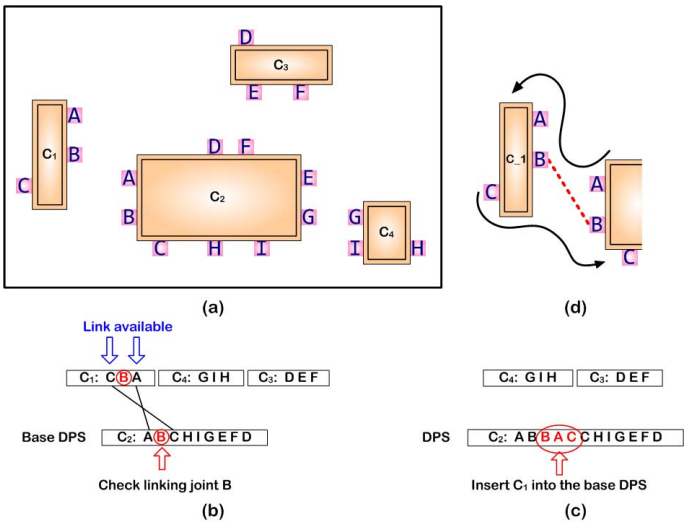

Fig. 3. An illustration of routing order determination by DPS. (a) is a simplified layout of PCB before routing. (b) shows an example of inserting C_1 into the DPS and determining the CCP. (c) shows that pin B in the DPS is decided as a CCP. Thus, C_1 is inserted into the DPS. (d) depicts that the connections of pin/net A and pin/net C can be constructed after the connection of pin B has been decided.

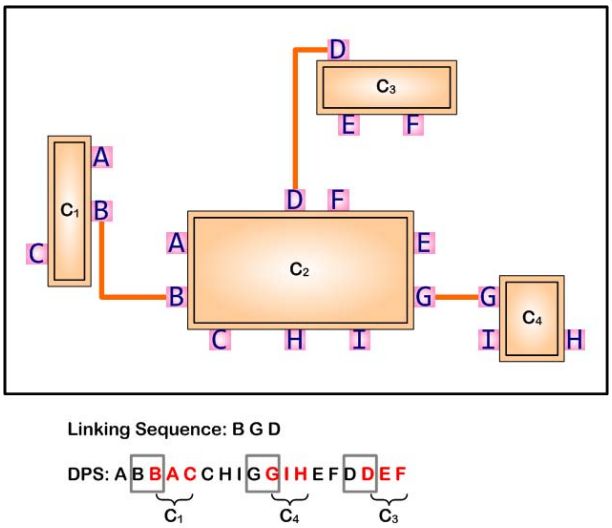

Fig. 4. An example of the first pin set BGD for routing. We then connect the adjacent pins according to the DPS and perform the consequent routing.

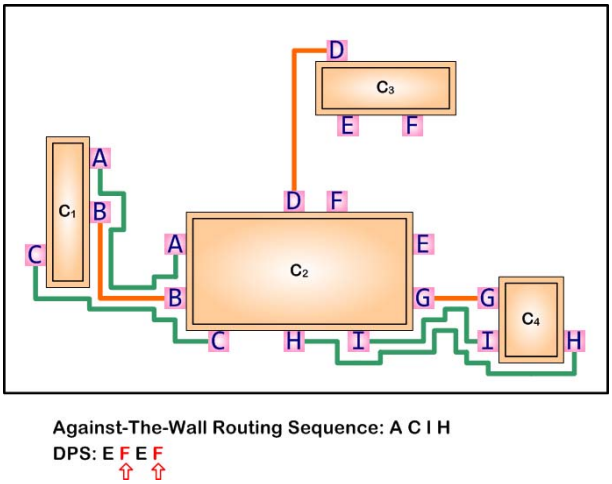

Fig. 5. The routing result of against-the-wall routing. All nets are connected in order and routed against the "walls". The results show that the nets will be close to the wires of the connection of the CCPs and have resembled wire shape for bus requirement.

can be applied to pin G and pin D in the DPS. The first set of erasure is pins B, G, and D, they act as the start point of topological routing, shown in Fig. 4 and presented in next subsection.

We can in fact choose any position/pin location as our CCP during each concatenation, however the choice itself will produce different configurations and then different ordering. This will possibly cause the crossing in some nets which could have been avoided. We therefore use a greedy-based approach to minimize the crossings for topological routing. For each concatenation, we evaluate each pin in the smaller pin-sequence (being concatenated) to check the number of erasures which can be performed. For example, in Fig. 3, for pin B, after the erasure of symbols BB, we can continue to erase CC and AA, therefore the topological routing for this subset can be performed without crossing. We then choose this pin with largest number of erasures to be the CCP. Moreover, we found that if the cyclic order (counterclockwise) of sub-pin-sequence is not the same, there are more than one pin which can be the candidate of the CCP. In component C_1 of Fig. 3, the pin order ACB is not the same as ABC in the sub-pin-sequence of component C_2. So pins A, B, and C can be the CCP. On the other hand, we observe that if the cyclic order of those sub-pin-sequences are the same, there will be crossings.

The advantages of this static net ordering are as follows. First, through the steps of pin-sequence concatenation, we can decide the "backbone" of the topological routing. Second, we use this ordering in order to preserve the routing space so that it will not be split to many pieces, avoiding the possible crossing during sequential routing, especially if the shortest path or mazing routing is used. Third, we have a considerably good static order for routing nets, compared with blind or dynamically changed orders.

B. Against-the-Wall Topological Routing

The DPS in Fig. 4 shows that we have the first routing set (collection of CCPs) in this example: *BGD*. Here we perform the routing to connect those pins. A^* algorithm [3] [7] is used in our router to complete the connections. A^* algorithm connects the net by calculating the path cost. Fig. 4 shows the result of the connections.

In order to fulfill the requirement of high speed board signal routing, we intend to route them as close as possible and consider the length difference constraints. Given the routing shown in Fig. 4, the routing space is split into three pieces. By using the topologically routed nets as boundaries/walls, we route the following nets "against-the-wall". Following the previous discussion, the erasure of symbol pairs in the DPS will produce the next sets of topological routing. We repeat this process until no symbol pairs existed in the DPS. For instance, in Fig. 4, after the routing of the sequence BGD, the DPS becomes $AACCHIIHEFEF$. Therefore the next

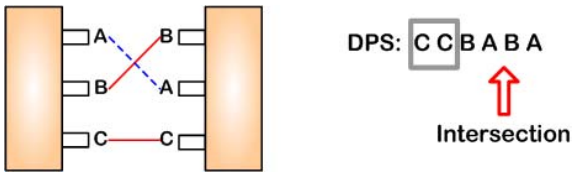

Fig. 6. A case which has inevitable net crossing in planar routing.

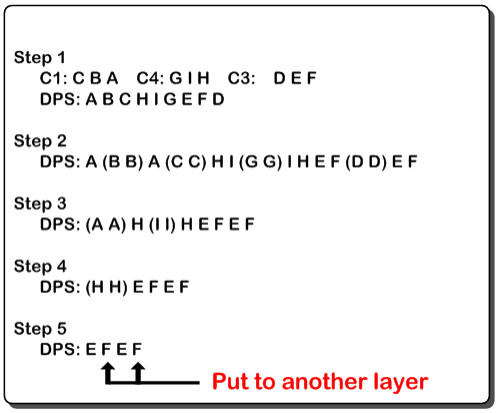

Fig. 7. An example to show the whole process of global topological routing. First, all components are seen as pin-sequences. Next, these pin-sequences are combined as one DPS. Then, the same adjacent pins are connected by the against-the-wall routing process. Finally, some pins cannot be erased in DPS, we can assign them to another layer for routing.

set of routing sequence is ACI. The DPS then becomes $HHEFEF$. The last set of routing is H. The final against-the-wall routing is shown in Fig. 5. In another point of view, we can treat the first routed nets as gravity centers, and the rest of nets are attracted to be placed besides them. We will discuss how we handle the rest of the DPS in next subsection.

We briefly describe the against-the-wall routing as follows. By using routed nets as boundary of routing space, the routes walk against the wall/boundary. We follow the order we describe previously to route the rest of nets, each route has a fixed direction towarding the destination pin. If there exist blockages (or previously routed nets), the direction will be modified to walk around. If there are more than two components which connnect to each other, after generating three CCPs, we have a triangle. In this case, we can still use the individual CCP connection route as a wall to guide the consequent routing.

C. Handling the Crossing Nets

It is possible that the design may not be routable in one layer based on our method, as shown in Fig. 6. We can in fact arrange some routes to be routed in other layers; it is also mentioned in [5]. We choose the pin/net that will cause the largest number of crossings in the current layer and assign it to another layer. Fig. 7 shows the complete steps to perform the global topological routing.

IV. Length-Constraint-Aware Routing Refinement

After the step of the against-the-wall routing, our router produces a series of bus nets. In this stage we adjust the routing without changing the completed interconnection in order to match the length constraints on the board, to resemble the net shape and to shorten (sometimes lengthen) the wire-length for longer (shorter) nets. Recall that during the process of the against-the-wall routing, a central ("backbone") net is generated and the other nets are routed close to the central net, thus more unused area can be reserved for later use. In the process of the net/bus adjustment, the adjusting order follows the reverse order of against-the-wall routing sequence. The reason for using such order is that, based on our experience, the latest net route length is usually much longer and exceeds the length constraint. For instance, pin/net H in Fig. 5 is likely to violate the length constraint, and it is also the latest routed net in the routing sequence.

For the efficiency of the net adjustment, we use a number sequence to represent the net segments. As shown in Fig. 8(a), we define the four basic directions, rightward, upward, leftward and downward as the numbers 1, 2, 3, 4 in counterclockwise fashion. Each net is divided into several segments and represented with these directions. Each segment has one head point, one tail point and one direction. We use this segment sequence as a pattern to optimize and adjust the routed length to fit the length constraint. Fig. 8(b) is an example of a segment sequence. During the process of net length adjustment, we recognize this pattern and apply some actions to shorten the length, one option is to move the segment 4 in the middle of the sequence to shift to left. Besides the pattern (segment sequence), additional costs for each segment are evaluated to determine which segments are suitable for length adjustment, it consists of several parts. First, we favor the shorter segments to be moved/shifted since longer segment wires are harder to adapt with the net length adjustment. Second, we check if there are more available spaces for the segments to move. More free segments get higher priority to be moved/shifted. Third, if other neighboring segments have no free space to move, the targeted segment is hard to be moved as well. We use the degree of the flexibility of the neighboring segments as part of the segment's cost. The segments with costs for shift priority are put into a shifting-list which records the order of shifting segments.

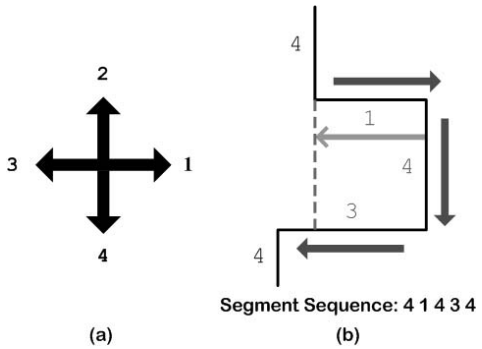

Fig. 8. An illustration of formulating the segment sequence. (a) is the four basic directions of the segments. (b) is a net which is transformed into segment sequence 41434.

During the segment shifting process, we use the following terms to control the net length. The first one is the average of total net length (Avg) among the set of nets. The second one is the length difference ($Diff$) which is the absolute difference between the certain net and the length average. The last one is the matching ratio used to limit the difference of the net in certain range ($Range$). In general, the matching ratio ($Range = |Avg - Diff|/Avg$) should be higher than 95%.

When processing net shifting, each net has their adjusting region which is the free space for the net shifting. Each net can adjust its length in the adjusting region to fit the length constraints. As the net shifting process begins, a segment is chosen to be moved in the adjusting area. The adjustment must follow the constraints which refer to the length average, length difference and matching ratio. If the moving segment finishes shifting, it is removed from the shifting-list. Then we choose another segment in the shifting-list and perform the iterative moving process. If the moving net has no segments in the shifting-list or the wire length of this moving net conforms with the length constraints, the moving process of one net is finished. In order to cease the whole net shifting process, we will stop if there is no improvement of the net shifting.

The benefits of the moving process include the available area release for the next moving segment. Fig. 9(a) shows a part of net before net shifting process and Fig. 9(b) shows the net after performing the net shifting process. This process is applied until the original head is overlapped with the new head (Fig. 9(b)) or the original tail is overlapped with the new tail. Against-the-wall routing is again used here to go around the obstacles.

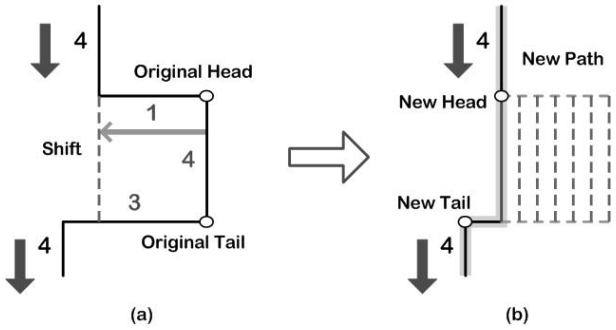

Fig. 9. An example of net shifting. (a) shows a net path before moving process. (b) After shifting the net segment, the net length is balanced and the space occupied by the net will be released for matching the other nets.

It is possible to increase the wire length while performing the refinement since we need the wires to detour (snaking). When hitting the obstacles, detour is needed and the length of this net increases.

V. Routing Results

We have implemented our methodology in C++ and the platform is on Intel Dual Core 2.2GHz with 2GB memory. We have constructed the test cases which are based on cases shown in recent works/references (Test Cases I and II resemble the cases shown in [16] since the cases in [16] are not available; Test Cases III to V are from [1], provided from design houses). Table I shows the test cases and routing efficiency for our router. $Nets$ and $Component$ indicate the number of nets and components in test cases. $Grid\ size$ shows the size (the number of grid cell on PCB board)[3] of the routing problem, $Total\ Time$ includes the time for building the routing grid and the run time spent by our router.

Fig. 10 shows the Test Case I which has two components. However, the routing results are different from that of [16]. Our results show that almost all nets are routed in certain region and the routing area is better utilized. The top-left enlarged view shows our router can shift the nets in a certain region to meet the length constraints. The top-right enlarged view shows the nets are routed in the similar shape. Moreover, many net-joggings are avoided due to non-floorplan-based approach (area sizing).

[3]The grid sizes of the test cases in this work are larger than the BSG grid sizes in [16], which shows the granularity of our approach.

TABLE I
The summary of test cases and experimental results.

	#Nets	#Component	Grid size	Total Time
Test Case I	17	2	250 × 250	2 sec
Test Case II	18	4	300 × 230	2 sec
Test Case III	36	5	300 × 230	2 sec
Test Case IV	81	8	640 × 560	10 sec
Test Case V	104	9	900 × 900	31 sec

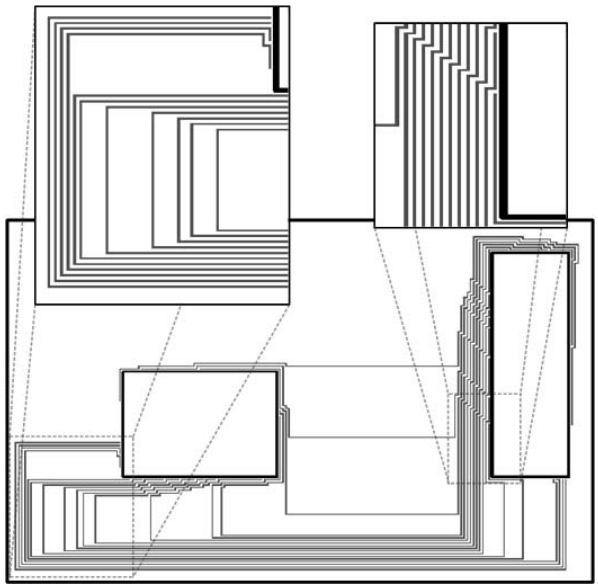

Fig. 10. The routing result of Test Case I. The result shows that the routing nets are centralized as buses and the routing area is better utilized. The top-left enlarged view shows our router can shift the nets in a certain region to meet the length constraints. The top-right enlarged view shows the nets are routed in the similar shape.

Fig. 11 shows Test Case III which has more components than the case in Fig. 10. The net distribution between each component are similar to the result of bus routing. Our router can also detour the nets to avoid crossing and conform with the length constraints. Even the components are not face to face, our router can still find a good solution. Fig. 12 (Test Case III with many obstacles) shows that our approach is more effective in handling obstacles. With our against-the-wall routing, we can go around the obstacles and the length constraints can be conformed.

In addition, we implement [4] to check the feasibility of bus router with obstacles. Although the problems are not quite the same, we can compare on this feasibility. Fig. 13 shows our implementation of [4] on this case. The circle shows that there exists no enough capacity for that bus to be successfully routed. In this case, we only show the bus in a bunch, the grouping of boundary pins for bus routing is not shown.

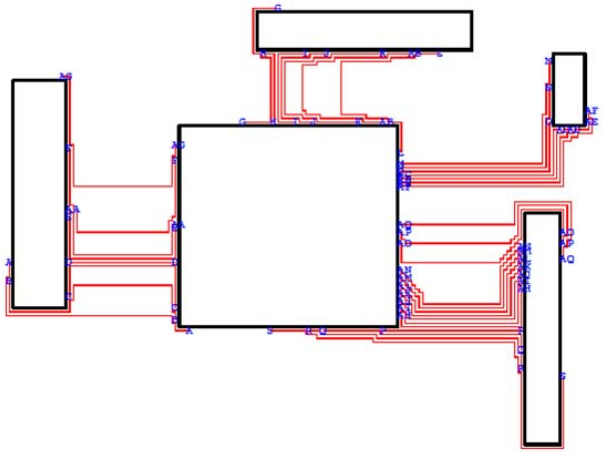

Fig. 11. The routing result of Test Case III. This case has five components and the length constraints are conformed.

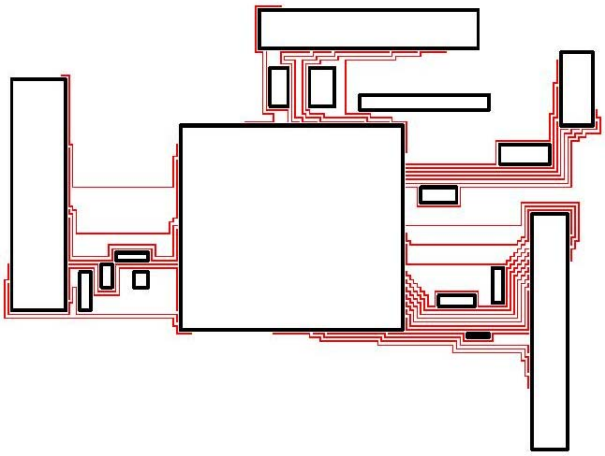

Fig. 12. The routing result of Test Case III with obstacles. In addition to original five components, we have inserted obstacles to show the effectiveness of our approach, and the runtime is same as that in Fig. 11.

VI. Conclusion

In this work, we have proposed a board router under fixed-ordering boundary pins. Unlike the conventional router which applies the shortest path concept, our router can better utilize the routing space, and consider the wire length and shape requirements. The experimental results show that we can preserve routing space in sequential planar routing under given boundary pin assignment and conform with the length constraints.

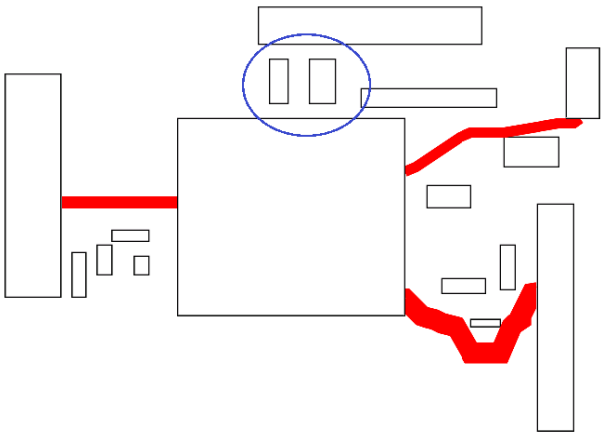

Fig. 13. The routing result of Test Case III with obstacles by our implementation of [4]. The circle shows that there exists no enough capacity for that bus to be successfully routed.

References

[1] "Taiwan MOE IC/CAD Contest". 2009.

[2] W.-M. Dai, T. Dayan, and D. Staepelaere. "Topological Routing in SURF: Generating a Rubber-Band Sketch". In *Proceedings of ACM/IEEE Design Automation Conference*, pages 39–44, 1991.

[3] Z. Dong and M. Li. "A Routing Method of Ad Hoc Networks Based on A-star Algorithm". In *International Conference on Networks Security, Wireless Communications and Trusted Computing*, pages 623–626, 2009.

[4] H. Kong, T. Yan, and D.-F. Wong. "Automatic Bus Planner for Dense PCBs". In *Proceedings of ACM/IEEE Design Automation Conference*, pages 326–331, 2009.

[5] H. Kong, T. Yan, and D.-F. Wong. "Optimal Simultaneous Pin Assignment and Escape Routing for Dense PCBs". In *Proceedings of IEEE Asia and South Pacific Design Automation Conference*, pages 275–280, 2010.

[6] Y. Kubo, H. Miyashita, Y. Kajitani, and K. Tateishi. "Equidistance Routing in High-Speed VLSI Layout Design". In *Proceedings of the Great Lakes Symposium on VLSI*, pages 439–449, 2005.

[7] P. Lester. "A* Pathfinding for Beginners". In *GameDev.net (http://www.gamedev.net/reference/articles/article2003.asp)*, July 2005.

[8] L. Luo and M. Wong. "Ordered Escape Routing Based on Boolean Satisfiability". In *Proceedings of IEEE Asia and South Pacific Design Automation Conference*, pages 244–249, 2008.

[9] L. Luo, T. Yan, Q. Ma, D.-F. Wong, and T. Shibuya. "B-Escape: A Simultaneous Escape Routing Algorithm Based on Boundary Routing". In *Proceedings of ACM International Symposium on Physical Design*, pages 19–25, 2010.

[10] Q. Ma, T. Yan, and M. Wong. "A Negotiated Congestion based Router for Simultaneous Escape Routing". In *Proceedings of International Symposium on Quality Electronic Design*, pages 606–610, 2010.

[11] M.-M. Ozdal and D.-F. Wong. "Simultaneous Escape Routing and Layer Assignment for Dense PCBs". In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, pages 822–829, 2004.

[12] M.-M. Ozdal and D.-F. Wong. "Algorithmic Study of Single-Layer Bus Routing for High-Speed Boards ". In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, volume 25, pages 490–503, Mar. 2006.

[13] M.-M. Ozdal and D.-F. Wong. "Two-Layer Bus Routing for High-Speed Printed Circuit Boards". In *ACM Transactions on Design Automation of Electronic Systems*, volume 11, pages 213–227, Jan. 2006.

[14] M.-M. Ozdal, D.-F. Wong, and P.-S. Honsinger. "An Escape Routing Framework for Dense Boards with High-Speed Design Constraints". In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, pages 759–766, 2005.

[15] M.-M. Ozdal, D.-F. Wong, and P.-S. Honsinger. "Simultaneous Escape-Routing Algorithms for Via Minimization of High-Speed Boards". In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, volume 27, pages 84–95, Jan. 2008.

[16] T. Yan and D.-F. Wong. "BSG-Route: A Length-Matching Router for General Topology". In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, pages 499–505, 2008.

[17] T. Yan and D.-F. Wong. "A Correct Network Flow Model for Escape Routing". In *Proceedings of ACM/IEEE Design Automation Conference*, pages 332–335, 2009.

[18] M.-F. Yu and W.-M. Dai. "Single-Layer Fanout Routing and Routability Analysis for Ball Grid Arrays". In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, pages 581–586, 1995.