

論文 / 著書情報
Article / Book Information

Title	A Fast Longer Path Algorithm for Routing Grid with Obstacles using Biconnectivity based Length Upper Bound
Authors	Yukihide Kohira, Suguru Suehiro, Atsushi Takahashi
Citation	IEICE Trans. Fundamentals, Vol. E92-A, No. 12, pp. 2971-2978
Pub. date	2009, 12
URL	http://search.ieice.org/
Copyright	(c) 2009 Institute of Electronics, Information and Communication Engineers

A Fast Longer Path Algorithm for Routing Grid with Obstacles Using Biconnectivity Based Length Upper Bound**

Yukihide KOHIRA^{†a)}, Member, Suguru SUEHIRO^{††*}, Nonmember, and Atsushi TAKAHASHI^{†††}, Member

SUMMARY In recent VLSI systems, signal propagation delays are requested to achieve the specifications with very high accuracy. In order to meet the specifications, the routing of a net often needs to be detoured in order to increase the routing delay. A routing method should utilize a routing area with obstacles as much as possible in order to realize the specifications of nets simultaneously. In this paper, a fast longer path algorithm that generates a path of a net in routing grid so that the length is increased as much as possible is proposed. In the proposed algorithm, an upper bound for the length in which the structure of a routing area is taken into account is used. Experiments show that our algorithm utilizes a routing area with obstacles efficiently.

key words: routing design of PCB, longer path algorithm, upper bound of wire length

1. Introduction

In recent VLSI systems, due to the increase of operation frequency, signal propagation delays are requested to achieve the specifications with very high accuracy. In order to achieve the severe requirements, signal propagation delay is taken into account in the routing design of PCB (Printed Circuit Board). The specifications on routing design of PCB would be determined by the design of other parts. The signal propagation delay of a net consists of gate delay and routing delay, and depends on various parameters. However, in the routing design of PCB, the controllability of wire length is often focused on since it enables us to control the routing delay. If a routing method has the higher controllability on wire length, the routing delay would be controlled with higher accuracy. The length control is easy if the large area is reserved for route of a net. However, the routing area is usually limited, and shared by multiple nets. Therefore, a routing method should utilize the routing area with obstacles as much as possible in order to realize the specifications of multiple nets simultaneously. In the literatures, the problem formulations such as equal-delay or equal-length routings

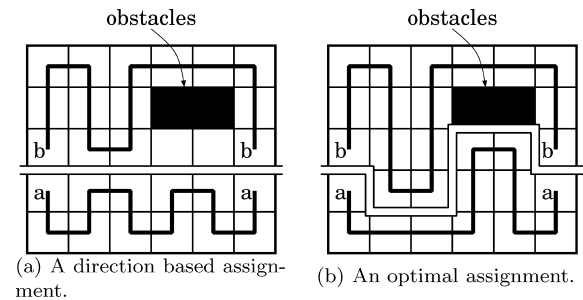


Fig. 1 Routing area assignment.

for multiple nets were often used to demonstrate the ability of methods.

For the routing problem for multiple nets, the methods were proposed recently [3], [4]. In [3], a part of routing area is assigned to each net according to the direction of the net. In [4], a Lagrangian relaxation technique is used for length matching on multi-layers. These algorithms work well when the routing area does not have obstacles. However, the method proposed in [3] is not applicable to most PCB designs since obstacles are not taken into consideration. Although the method proposed in [4] is applicable when the routing area has obstacles, it is not guaranteed to obtain a feasible solution.

Assume that net *a* and *b* are assigned routing areas as shown in Fig. 1(a) and Fig. 1(b), and a routing method tries to utilize the assigned routing area as much as possible. The sizes of the assigned areas for each net are the same in both figures, and the wire lengths of net *a* are the same in both figures. However, the wire lengths of net *b* differ. The existence of obstacles should be taken into account in routing area assignment.

The size of a routing area is not enough to evaluate the area. In order to evaluate a routing area with obstacles, the maximum wire length that can be achieved within the area should be known. However, it is an NP-hard problem since its decision version is NP-complete [5]. Therefore, a fast heuristic algorithm that evaluates a routing area with obstacles by the achievable wire length is desired.

In this paper, first, an upper bound for the wire length of a net in a routing grid with obstacles is proposed. The proposed upper bound uses the concept of the biconnected component to exclude useless areas for routing efficiently. Then, a longer path algorithm that generates the route of a net so that the wire length is increased as much as possible

Manuscript received March 19, 2009.

Manuscript revised June 17, 2009.

[†]The author is with the School of Computer Science and Engineering, the University of Aizu, Aizu-Wakamatsu-shi, 965-8580 Japan.

^{††}The author is with the Department of Communications and Integrated Systems, Tokyo Institute of Technology, Tokyo, 152-8550 Japan.

^{†††}The author is with the Graduate School of Engineering, Osaka University, Suita-shi, 565-0871 Japan.

*Presently, with Nomura Research Institute, Ltd.

**The preliminary versions were presented at [1], [2].

a) E-mail: kohira@u-aizu.ac.jp

DOI: 10.1587/transfun.E92.A.2971

is proposed. In the proposed algorithm, the route of a net is greedily determined by using the proposed upper bound. In experiments, we show that a routing grid with obstacles is highly utilized by our proposed algorithm, and that the proposed upper bound helps our proposed algorithm in generating a longer route. To the best of our knowledge, there is no good heuristic algorithm in comparison. Several heuristic algorithms are also introduced and used in comparison.

2. Preliminaries

In this paper, a routing area with single layer is represented by a routing grid consisting of unit squares, and a route is represented by thick line consisting of horizontal and vertical line segments as shown in Fig. 2. In the following, a unit square that corresponds to an obstacle is drawn by black in figures, and called as *off-square*. The other grids are drawn by white or gray and called as *on-squares*. From the on-squares, two squares are specified as a *start* and a *goal* of a route of a net. A route connects the start square and the goal square by line segments. A route passes each on-square at most once and must not pass any off-squares. A start square and a goal square are denoted by a single circle and a double circle in figures, respectively. The *length* of a route is the number of squares of the route including its start and goal squares.

A planar routing area is represented by G for simplicity. Note that an arbitrary routing area can be represented by setting obstacles on the boundaries. Each square of a routing grid is referred by their coordinate. The coordinate of the leftmost-bottom square is defined as (1, 1). In the following, graph terminologies are used in explanation by assuming a grid graph is defined from a routing grid, but a routing grid representation is used in figure for simplicity and readability.

Let $L_{\max}(G, s, t)$ be the length of a longest route which connects start square s and goal square t by using on-squares in G only. If it is not confusing, $L_{\max}(G, s, t)$ is denoted by L_{\max} and the length of a route is denoted by L for simplicity. An example is shown in Fig. 2. The start and goal squares are (1, 1) and (1, 6), respectively. The length of the route is 10 ($L = 10$).

The problem we concern is to find a route of a single net with L_{\max} . Methods for this problem are used as a subroutine of routing system or used to evaluate the area

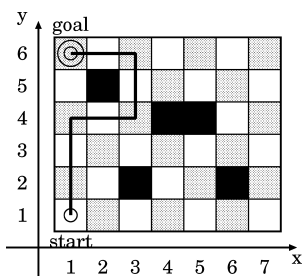


Fig. 2 Routing grid G_1 and a route ($L = 10$).

that will be assigned to a net. The problem to find a route with the minimum length in a grid graph with obstacles can be solved by maze routing in polynomial time [6]. On the other hand, it is known that Hamiltonian path problem in a grid graph with obstacles is NP-complete [5]. Since Hamiltonian path problem is a decision version of the problem to find a route with L_{\max} , the problem to find a route with L_{\max} is NP-hard. Therefore, we investigate heuristic algorithms for this problem and upper bounds of length of a route.

3. Upper Bounds of Length

In this section, we introduce three upper bounds of length.

3.1 Upper Bound by Bipartition

Since a grid graph is bipartite, the set of squares is divided into two subsets, white and gray, so that any two squares in a subset are not adjacent to each other (see Fig. 2). Then, any route passes white and gray squares alternatively. Therefore, some on-squares of a subset can not be used in a route if the on-squares in the other subset are fewer than the on-squares in the subset. This analysis gives an upper bound of length of a route [7]. We assume that there is a path from a start square to a goal square.

Theorem 1 ([7]): Let G be a grid graph, and s and t be on-squares in G . Let V_w (V_g) be the set of white (gray) on-squares in G . Let $U_p(G, s, t)$ be

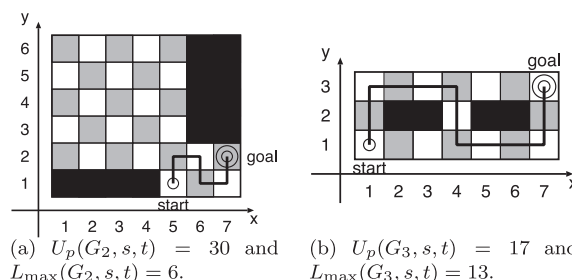
$$U_p(G, s, t) = \begin{cases} 2 \min\{|V_w| - 1, |V_g|\} + 1 & (\text{if } s, t \in V_w) \\ 2 \min\{|V_w|, |V_g| - 1\} + 1 & (\text{if } s, t \in V_g) \\ 2 \min\{|V_w|, |V_g|\} & (\text{otherwise}). \end{cases}$$

$U_p(G, s, t)$ is an upper bound of length of a route that connects s and t in G . \square

$U_p(G, s, t)$ is denoted by U_p for simplicity. The upper bound U_p is easy to calculate, but not tight in general. Examples in which U_p is larger than L_{\max} are shown in Fig. 3.

3.2 Upper Bound by Biconnected Components

In this section, an upper bound in which the structure of a



(a) $U_p(G_2, s, t) = 30$ and $L_{\max}(G_2, s, t) = 6$. (b) $U_p(G_3, s, t) = 17$ and $L_{\max}(G_3, s, t) = 13$.

Fig. 3 Routing grid G_2 and G_3 .

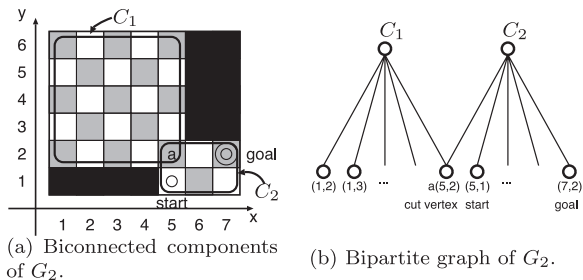


Fig. 4 Biconnected components of G_2 and its bipartite graph.
 $U_c(G_2, s, t) = 6$.

routing grid is taken into account is introduced. By using the concept of biconnected component, a tighter upper bound is obtained efficiently.

A graph is said to be *biconnected* if it is connected even if any one vertex is removed from it. Note that a graph which consists of two vertices connected by an edge is biconnected. A *biconnected component* of a graph is a maximal biconnected subgraph of the graph. A vertex of a connected graph is said to be a *cut vertex* if the graph obtained by removing the vertex is unconnected. In general, the graph has at least one biconnected component. Every cut vertex belongs to at least two biconnected components, and the others belong to exactly one biconnected component. It is known that the time complexity to enumerate biconnected components is linear to the sum of the number of vertices and edges [8]. For example, the number of biconnected components of G_2 shown in Fig. 3(a) is two, and biconnected components are shown in Fig. 4(a).

Here, we briefly show that, for any pair of vertices in G , the sequence of biconnected components of G that corresponds to the path between them is unique.

Let G_B be the bipartite graph obtained from a connected graph G as follows. A vertex of G_B corresponds to either a vertex in G or a biconnected component of G . A vertex belongs to a biconnected component in G if and only if an edge connects the corresponding vertices in G_B . Note that G has a path that connects two vertices if and only if G_B has a path that connects the corresponding vertices. For example, the bipartite graph of G_2 is shown in Fig. 4(b).

Suppose that G_B has a cycle. Since a vertex in G_B that corresponds to a vertex but not to a cut vertex in G has exactly one incident edge, the cycle contains a vertex v_c that corresponds to a cut vertex in G . Since v_c is on a cycle of G_B , the graph obtained from G_B by deleting v_c is connected. Also, the graph obtained from G by deleting the vertex corresponding to v_c is connected, and this contradicts the definition of cut vertex. Therefore, G_B has no cycle and G_B is a tree. In a tree, a path connecting any two vertices is unique. From this observation, the sequence of biconnected components that corresponds to a route from the start square to the goal square is unique. A square that does not belong to a biconnected component in the sequence can not be used by any route from the start square to the goal square. This analysis gives an upper bound of length of a route.

Theorem 2: Let G be a grid graph, and s and t be on-squares in G . Let (C_1, C_2, \dots, C_n) be the sequence of biconnected components, and (c_0, c_1, \dots, c_n) be the sequence of cut vertices, such that $c_i \in C_i \cap C_{i+1}$ ($0 < i < n$), $c_0 = s$, and $c_n = t$. Let $U_c(G, s, t)$ be

$$U_c(G, s, t) = \begin{cases} 0 & \text{(if } G \text{ has no path between } s \text{ and } t) \\ 1 & \text{(if } s = t) \\ \sum_{i=1}^n U_p(C_i, c_{i-1}, c_i) - n & \text{(otherwise).} \end{cases}$$

$U_c(G, s, t)$ is an upper bound of length of a route which connects s and t in G . \square

For example, $U_c(G_2, s, t) = 6$ which is equal to $L_{\max}(G_2, s, t)$ while $U_p(G_2, s, t) = 30$ (see Fig. 4(a)). On the other hand, $U_c(G_3, s, t) = 17$ which is not equal to $L_{\max}(G_3, s, t) = 13$.

3.3 Proposed Upper Bound

The gap between the upper bound U_c defined in the previous section and the actual maximum length L_{\max} becomes small, but still quite large in some cases. The upper bound can be improved by adopting a lookahead technique. That is, when a partial route from the start square or the goal square is generated, the sum of the upper bound of length of the remaining and the length of the generated partial route is an upper bound of length with the constraint such that the partial route is fixed. If all the possibilities of partial routes are enumerated, the actual maximum length does not exceed the maximum of them. That is, the maximum of them gives an upper bound of length. A tighter upper bound is obtained if the degree of lookahead is increased. However, there is a trade-off between the accuracy and the computational time. As the degree of lookahead is increased, the number of combinations explodes and the computation time increases exponentially.

In our proposed method, all the partial routes of length one from both a start square and a goal square are generated. Then, all the combinations are evaluated and the maximum of the obtained values is used as an upper bound of length by lookahead. $U_a(G, s, t)$ which is defined in the following is used as a tighter upper bound of length of a route.

Theorem 3: Let G be a grid graph, and s and t be on-squares in G . Let $A(v)$ be the set of on-squares which adjacent to v in G , and $G/\{s, t\}$ be the grid graph obtained from G by removing s and t . Let $U_a(G, s, t)$ be

$$U_a(G, s, t) = \begin{cases} 0 & \text{(if } G \text{ has no path between } s \text{ and } t) \\ 1 & \text{(if } s = t) \\ 2 + \max_{s' \in A(s), t' \in A(t)} U_c(G/\{s, t\}, s', t') & \text{(otherwise).} \end{cases}$$

$U_a(G, s, t)$ is an upper bound of length of a route which connects s and t in G . \square

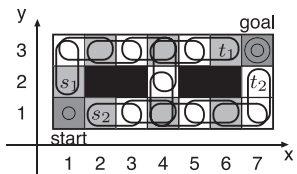


Fig. 5 Biconnected components of $G_3/\{s, t\}$. $U_a(G_3, s, t) = 13$.

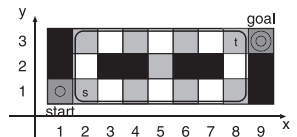


Fig. 6 An example in which U_a has an error.

For example, $U_a(G_3, s, t) = 13$, since upper bound $U_c(G_3/\{s, t\}, s_1, t_2) = 11$ and it is maximum among all combinations (see Fig. 5).

Here, we discuss the time complexity in computing U_c and U_a . Let V be the set of squares in a grid graph G , and let E be the set of edge in G . Since $|E|$ is $O(|V|)$ in G , the time complexity of obtaining biconnected components is $O(|V|)$ [8]. Therefore, the time complexity in computing U_c is $O(|V|)$. Since the number of neighbors of a square is at most four, the time complexity in computing U_a is also $O(|V|)$.

There exist instances in which our proposed upper bound U_a contains errors. For example, the gap between U_a and the longest path in the instance shown in Fig. 6 is 4. Note that the instance shown in Fig. 3(b) is obtained if the partial routes of length one from both a start square and a goal square are generated in the instance shown in Fig. 6.

4. Proposed Routing Method

In this section, we propose a fast longer path algorithm that generates a route of a net so that the length of the route is increased as much as possible. The proposed algorithm uses the upper bound U_a as a preferred function. We call our proposed routing method *US routing* (Upper bound based Seed routing).

In US routing, the route is determined by moving the *frontier* from the start square to the goal square. The frontier on a square is moved to an adjacent on-square such that U_a after the movement is maximum.

More precisely, the movement of the frontier in finding a route from s to t in G is described in the following. Assume that the frontier is on v . Let G' be the grid graph obtained from G by changing squares on the partial route from s to v defined by the movement of the frontier to off-square. Let $A(v)$ be the set of on-squares which are adjacent to v in G' . Let u be a square in $A(v)$ such that $U_a(G', u, t)$ is maximum among squares in $A(v)$. Then the frontier on v is moved to u in US routing.

Ties are broken by selecting a square such that the distance from the goal square t in G' is maximum, then broken

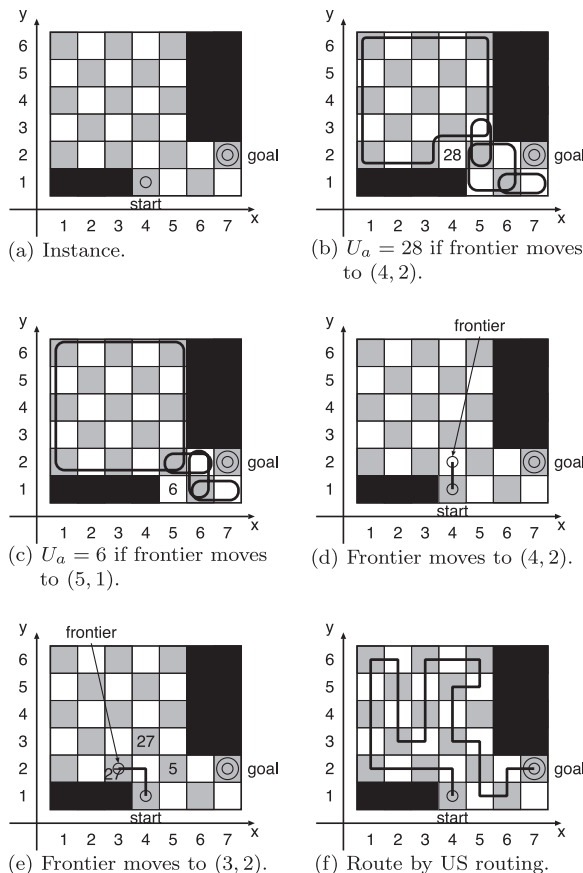


Fig. 7 An example of behavior of US routing.

arbitrarily. This tie breaking is used as preferred function in Furthest routing described in the next section.

An example of behavior of US routing is shown in Fig. 7. If frontier moves from start (4, 1) to (4, 2), U_a is equal to 28 (Fig. 7(b)). On the other hand, if frontier moves to (5, 1), U_a is equal to 6 (Fig. 7(c)). Then, the frontier moves to (4, 2) (Fig. 7(d)). Next, the frontier moves to (3, 2) because of the tie breaking by the distance (Fig. 7(e)). The move of the frontier is repeated, and finally, the route shown in Fig. 7(f) is obtained.

US routing finds a route from a source square to a goal square if it exists. If the movement of the frontier forces a route not to reach the goal square, then U_a corresponding to the movement is 0. Such movement is never adopted if a route from a source square to a goal square exists since a movement whose U_a is positive always exists.

Here, we discuss the time complexity of US routing. The frontier arrives at the goal square by at most $|V|$ movements. In each movement, the number of computation of the upper bound U_a is at most four. Therefore, since the time complexity of U_a is $O(|V|)$, the time complexity of US routing is $O(|V|^2)$.

5. Routing Methods for Comparisons

In this section, several heuristic algorithms are introduced to

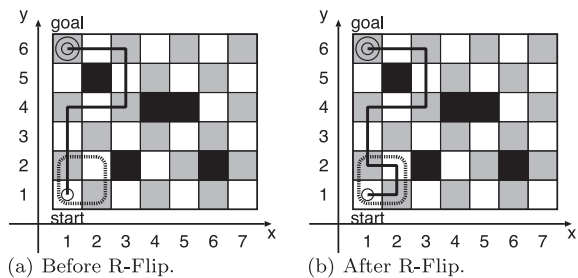


Fig. 8 R-Flip.

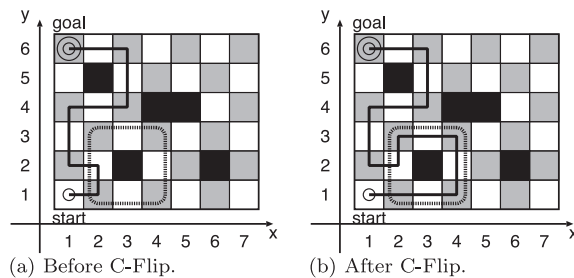


Fig. 9 C-Flip.

use them in comparison. Algorithms consist of two stages. At the first stage which is called *seed stage*, a route which connects the start and goal squares is generated. At the second stage which is called *expansion stage*, a route obtained by the first stage is modified greedily to increase the length of the route.

5.1 Seed Stage

In the seed stage, a route which connects the start and goal squares is generated by using one of three routing methods, “US,” “Maze,” or “Furthest.” US routing was explained in the previous section. The other two routing methods are explained briefly.

In Maze routing [6], the frontier is repeatedly moved to an adjacent on-square such that the distance from the goal square is minimum. The distance from the goal square is the length of a shortest path from the goal square using on-squares only. In Maze routing, first, each square is assigned a distance label, and the frontier moves according to the distance label without updating distance labels. It is well-known that a route with the minimum length is obtained by Maze routing and that the time complexity of Maze routing is $O(|V|)$ [6].

In Furthest routing, the frontier is repeatedly moved to an adjacent on-square such that the distance from the goal square is maximum. In contrast with Maze routing, distance labels are updated after each movement of the frontier since the distance from the goal square might be changed by a movement of the frontier. Of course, a route with the maximum length is not necessarily obtained by Furthest routing. The time complexity of Furthest routing is $O(|V|^2)$, since the time complexity of updating the distance label is $O(|V|)$ and the number of movements of the frontier is at most $|V|$.

5.2 Expansion Stage

In the expansion stage, a route obtained by the seed stage is modified greedily to increase the length of the route. *R-Flip* and *C-Flip* are introduced as the methods used in the expansion stage.

In R-Flip, a partial route of length two is detoured by using two adjacent on-squares (see Fig. 8). That is, a partial route of length two is replaced by a partial route of length four. In R-Flip, a partial route in four squares in a 2 times

2 rectangle such that a partial route passes two of them and that the others are on-square is modified. In our implementation, R-flip is applied greedily by searching an appropriate rectangle along the route from the start square to the goal square.

C-Flip is a generalization of R-Flip. A partial route R in the route is replaced by another partial route R' that passes on-squares and that has the same terminals of R . C-Flip is applied if the length of R is larger than the length of R' so that the length of the route increases (see Fig. 9). To find a candidate of C-Flip, each square is assigned a label that corresponds to the length of a shortest path from a square v on the route that passes on-squares. If the label of a square on the route is larger than the length of a partial route from v , C-Flip is applied. In our implementation, such pair is searched by changing the start square v of a partial route from the start square.

In [9], Flip is proposed as a route modification method. Flip modifies the route according to a face on the plain graph. Since the rectangle with width two and length two is a face, R-Flip is a kind of Flip. On the other hand, C-Flip might not be a kind of Flip since the cycle defined by C-Flip might not be a face. C-Flip is introduced as a specialized method for increase of the length of a route.

6. Experimental Results

We implement the computation of upper bounds and the routing methods in C++, which compiled gcc4.1.2, and executed on a PC with 2.93 GHz Intel Core 2 CPU and 2 GB RAM. We apply each method to 5 artificially generated data (from Data1 to Data5), 5 data generated by referring to industrial design (from Data6 to Data10), and 4 random data such that start, goal, and obstacles were placed randomly.

First, we discuss the results of 10 sample data. The results of 10 sample data are shown in Table 1. Examples of obtained routes are shown from Fig. 10 to Fig. 15.

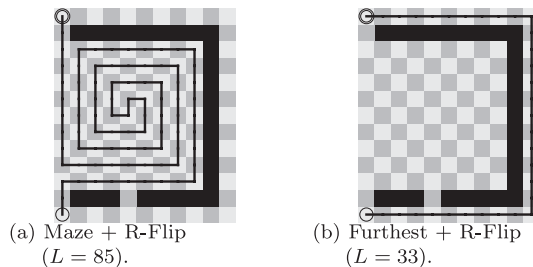
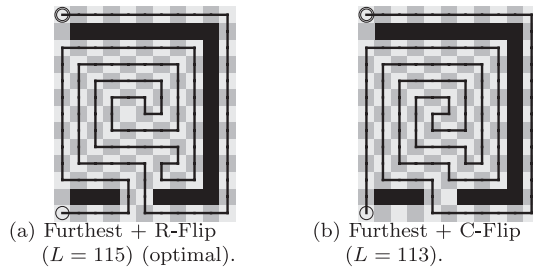
Every upper bound is less than the number of on-grids for every sample data except Data10. From Data1 to Data8, $U_c = U_p$ since the number of biconnected components is one. For Data9 and Data10, $U_c < U_p$ since the dead ends are excluded in U_c . From Data6 to Data8, $U_a < U_c$ since more than one biconnected component is generated by lookahead.

The gap between the length of route obtained by Maze routing or Furthest routing and the proposed upper bound

Table 1 Results of sample data.

	area	Upper bound				Maze routing			Furthest routing			US routing		
		#on	U_p [7]	U_c	U_a	+R-F.	+C-F.	+R-F.	+C-F.	+R-F.	+C-F.			
Data1	10x9	83	82	82	82	10	78	80	78	80	80	80	—	—
Data2	11x13	121	119	119	119	23	119	119	105	115	115	119	—	—
Data3	11x13	110	109	109	109	13	77	107	105	107	107	107	—	—
Data4	11x13	106	105	105	105	13	37	103	99	103	103	103	—	—
Data5	11x13	98	97	97	97	13	37	95	91	95	95	95	—	—
Data6	11x13	116	115	115	85	13	85	33	33	—	—	85	—	—
Data7	11x13	117	115	115	113	13	85	113	33	—	113	113	—	—
Data8	11x13	118	117	117	115	13	89	113	33	115	113	115	—	—
Data9 ([s])	20x20	297	291	267	267	9	—	231	251	—	261	251	—	—
Data10 ([s])	70x100	6654	6654	6650	6650	154	6584	6594	6490	6566	6566	6626	6628	6628
									1.21	1.21	1.21	4.76	4.80	4.80

#on the number of on-squares
 +R-F. the length of the route obtained by R-Flip after seed stage
 +C-F. the length of the route obtained by C-Flip after seed stage
 — means that the length is not increased by the expansion method.
bold figure means that the length of the route is equal to the upper bound.
 * Computation time which is less than 0.01[s] is omitted.

**Fig. 10** Routes for Data7.**Fig. 11** Routes for Data8.

U_a is large, since Maze routing and Furthest routing are not appropriate for the maximum length problem. Although the length of the route is improved at the expansion stage, the gap is still large in most data. The best combination of methods of seed stage and expansion stage depends on data. In average, Furthest routing is better than Maze routing as the method of seed stage and C-Flip is better than that R-Flip as the method of expansion stage. Examples of routes obtained by Maze routing or Furthest routing are shown in Fig. 10 and Fig. 11.

While, US routing obtains a longer route such that the route is not improved in the expansion stage in most cases (see from Fig. 12 to Fig. 15). Moreover, US routing is equal to or close to the proposed upper bound U_a . This fact im-

plies that the proposed upper bound U_a and US routing are promising.

US routing is heuristic. The length of the route obtained by US routing depends on the accuracy of the proposed upper bound. Since the proposed upper bound contains errors for several instances, the length of the route obtained by US routing is less than that by other routing methods in a few case (see Fig. 15). The computation time is long since the number of computations of biconnected components is large. These issues remain as future works.

Last, we discuss the results of 4 random data. Start grid and goal grid are placed randomly, each grid is randomly set as an obstacle by a certain ratio (0.1, 0.2, 0.3, or 0.4) in each data. The results of 4 random data are shown in Table 2 and examples of routes obtained by US routing are shown Fig. 16.

Note that the number of on-squares is decreased if the obstacle ratio is increased. Therefore, an upper bound and the length of a longer route are decreased by increasing the obstacle ratio. In 4 data, although $U_a = U_c$, U_a gives a tighter upper bound than U_p if the obstacle ratio is increased. If the obstacle ratio is increased, the computation time of each routing method is smaller since the number of on-squares are decreased. US routing obtains a longer route than the other methods in which Maze routing or Furthest routing is applied in the seed stage. However, in higher obstacle ratio, a longer route is obtained by applying R-Flip or C-Flip after applying US routing. This issue remains as a future work.

7. Conclusion

In this paper, we propose an upper bound for the wire length of a net in a routing grid with obstacles based on the biconnected component and a longer path algorithm US routing which greedily determines the route by using the proposed upper bound. In experiment, the lengths of routes obtained

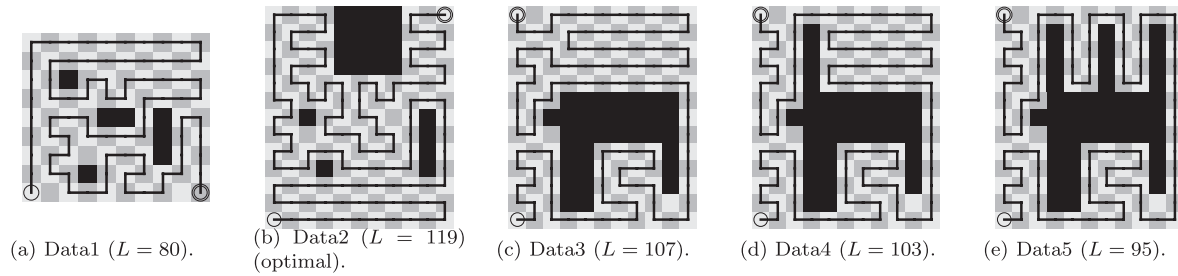


Fig. 12 Routes by US routing.

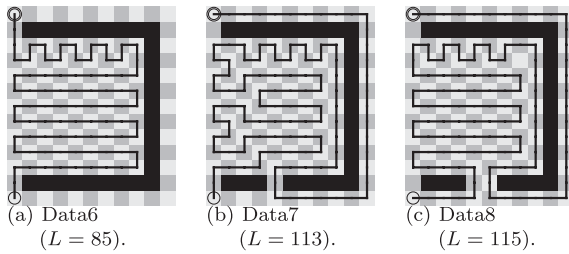


Fig. 13 Longest routes by US routing (optimal).

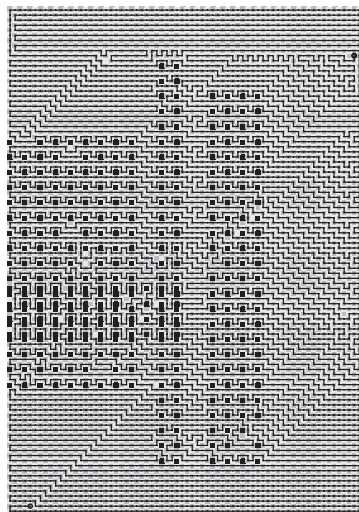


Fig. 14 Route for Data10 by US routing ($L = 6626$).

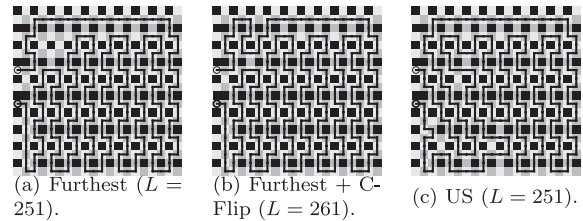


Fig. 15 Routes for Data9.

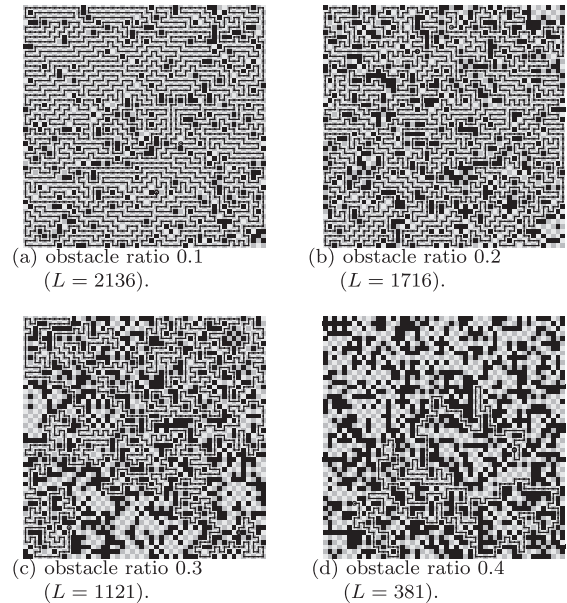


Fig. 16 Routes of US routing in random data.

Table 2 Results of random data such as start, goal, and obstacles are placed randomly in 50 times 50 rectangles.

obstacle ratio	Upper bound				Maze routing			Furthest routing			US routing		
	#on	U_p [7]	U_c	U_a	+R-F.	+C-F.		+R-F.	+C-F.		+R-F.	+C-F.	
0.1	2234	2200	2188	2188	16	1862	2096	2046	2100	2102	2136	2138	2138
								0.11	0.11	0.11	0.54	0.54	0.54
0.2	2010	1969	1853	1853	13	209	1725	1559	1719	1727	1761	1769	1769
								0.08	0.08	0.08	0.43	0.44	0.44
0.3	1754	1751	1305	1305	99	265	1149	1001	1087	1115	1121	1129	1147
								0.05	0.05	0.05	0.38	0.38	0.38
0.4	1505	1473	511	511	53	139	377	255	295	329	381	405	437
								0.01	0.01	0.01	0.18	0.18	0.19

upper figure upper bound or length
 lower figure computation time
 #on the number of on-squares
 +R-F. the length of the route obtained by R-Flip after seed stage
 +C-F. the length of the route obtained by C-Flip after seed stage
 * Computation time which is less than 0.01[s] is omitted.

by US routing are longer than those by compared methods in most data, and the upper bound is same or close to the wire length obtained by US routing. These results mean the proposed upper bound and US routing are promising.

In some cases, gaps between the proposed upper bound and the length of the route obtained by US routing are still large. The computation time of US routing is large since it computes our proposed upper bound iteratively. Improvement of proposed upper bound and improvements of US routing in terms of length and the computation time will be in our future works.

Our idea is applicable to practical PCB design. For example, assume that, after global routing is determined, the detailed routing of a net is determined to satisfy the wire length constraint one by one. In this case, an approximated detailed route is obtained by our method in which a routing area and grid is defined according to the design rules and the the global routing by estimating the required area. An adjustment of wire length that takes gridless and/or curve routing into account might be done in post processing. The detailed procedures are in our future works.

Acknowledgements

The authors would like to thank Toshiyuki Shibuya of Fujitsu Laboratories of America, Inc. for his helpful suggestions.

References

- [1] Y. Kohira, S. Suehiro, and A. Takahashi, "A maximum wire length router in routing area with obstacles using upper bound estimation considering connectivity," 21st Workshop on Circuits and Systems in Karuizawa, pp.569–574, 2008.
- [2] Y. Kohira, S. Suehiro, and A. Takahashi, "A fast longer path algorithm for routing grid with obstacles using biconnectivity based length upper bound," ASPDAC, pp.600–605, 2009.
- [3] M.M. Ozdal and M.D.F. Wong, "Algorithmic study of single-layer bus routing for high-speed boards," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.25, no.3, pp.490–503, 2006.
- [4] M.M. Ozdal and M.D.F. Wong, "A length-matching routing algorithm for high-performance printed circuit boards," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.25, no.12, pp.2784–2794, 2006.
- [5] A. Itai, C.H. Papadimitriou, and J.L. Szwarcfiter, "Hamiltonian paths in grid graphs," SIAM, Comput., vol.11, no.4, pp.676–686, 1982.
- [6] C. Lee, "An algorithm for path connections and its applications," IRE Trans. Electronic Comput., no.EC-10, pp.346–365, 1961.
- [7] S. Suehiro, Y. Kohira, and A. Takahashi, "An estimation of maximum wire length in routing area with obstacles," IEICE Technical Report, CAS2007–97, 2008.
- [8] R.E. Tarjan, "Depth-first search and linear graph algorithms," SIAM, Comput., vol.1, no.2, pp.146–160, 1972.
- [9] Y. Kubo, Y. Takashima, S. Nakatake, and Y. Kajitani, "Self-reforming steiner tree by flip and applications to VLSI interconnection," IPSJ Journal, vol.41, no.4, pp.881–888, 2000.



a member of IEEE and IPSJ.

Yukihide Kohira received his B.E., M.E., and D.E. degrees from Tokyo Institute of Technology, Tokyo, Japan, in 2003, 2005, and 2007, respectively. He had been a researcher of Department of Communications and Integrated Systems in Tokyo Institute of Technology from 2007 to 2009. He is currently with the School of Computer Science and Engineering, the University of Aizu, as an assistant professor since 2009. His research interests are in VLSI design automation and combinational algorithms. He is



Suguru Suehiro received his B.E. and M.E. degrees from Tokyo Institute of Technology, Tokyo, Japan, in 2007 and 2009, respectively. He is currently with Nomura Research Institute, Ltd. His research interests are in VLSI design automation and combinational algorithms.



Engineering, Graduate School of Engineering, Osaka University, as an associate professor since 2009. His research interests are in VLSI layout design and combinational algorithms. He is a member of IEEE and IPSJ.

Atsushi Takahashi received his B.E., M.E., and D.E. degrees in electrical and electronic engineering from Tokyo Institute of Technology, Tokyo, Japan, in 1989, 1991, and 1996, respectively. He had been with the Tokyo Institute of Technology as a research associate from 1991 to 1997 and had been an associate professor from 1997 to 2009. He visited University of California, Los Angeles, U.S.A., as a visiting scholar from 2001 to 2002. He is currently with Division of Electrical, Electronic and Information