A Fixed-Length Routing Method Based on the Color-Coding Algorithm

Tieyuan PAN
Graduate School of Environmental Engineering
University of Kitakyushu
Kitakyushu,Fukuoka,Japan
t2mcb008@eng.kitakyu-u.ac.jp

Yasuhiro TAKASHIMA
Faculty of Environmental Engineering
University of Kitakyushu
Kitakyushu, Fukuoka,Japan
takasima@kitakyu-u.ac.jp

Abstract— This paper proposes a fixed-length routing method based on the color-coding algorithm. In recent LSI system design, exact signal propagation delay is required because of the growth of the operation frequency. To control the delay, the fixed-length routing is widely used. This paper proposes a fixed-length routing method based on the color-coding algorithm. We analyze the complexity of the proposed approach and confirm its efficiency empirically.

I. Introduction

In recent LSI systems, the high accuracy of the signal propagation delay is required. This requirement is also considered in Printed Circuit Board (PCB) designs. The routing methods in [1, 2] minimize the total wire length of nets for PCB routing, but they do not consider the signal propagation delay. The signal propagation delay of the PCB is determined by a number of parameters such as gate-delay and routing-delay. It is difficult to adjust the signal propagation delay since the parameters depend on each other. To simplify the problem, the fixed-length routing problem is widely used. For the fixed-length routing problem, several research results [3–5] based on the Bus-Routing have been proposed without considering the obstacles. However, the consideration of obstacles in the routing area of PCB is much important in practical, since there are many obstacles in the routing area of PCB, i.e., IC packages and other devices.

In [6], CAFE router has been proposed, which is a fast connectivity aware multiple nets routing algorithm. It utilizes the network-flow algorithm. Thus, the CAFE router may obtain a feasible routing if the terminal positions satisfy the river routing condition. If the terminal positions do not satisfy the condition, the flow might connect the terminals of the different nets. A failure example is shown in Fig.1. In the figure, the vertices show the routing grids and the labeled one corresponds to the terminal of the net. The left terminal of the net B is connected to the right terminal of the net C. Thus, this routing of the figure is not feasible. To avoid this infeasibility, [6] proposed that some *walls* are constructed. An example of wall construction is shown in Fig.2. In this figure, the walls are shown by the thick lines. The wall construction achieves the feasible routing.

However, the wall construction still remains a difficult problem. An example is shown in Fig.3. In this figure, the walls

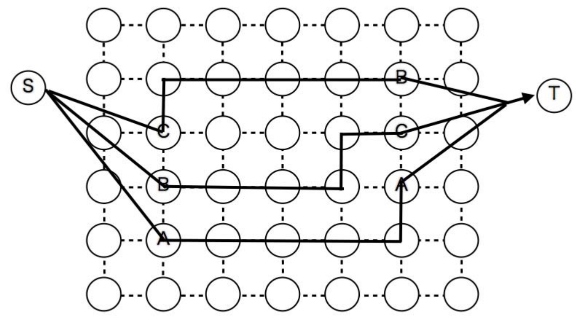

Fig. 1. An example of the violation of river routing condition

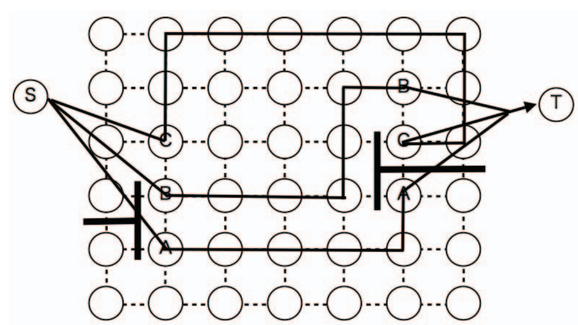

Fig. 2. An example for the wall construction in CAFE router

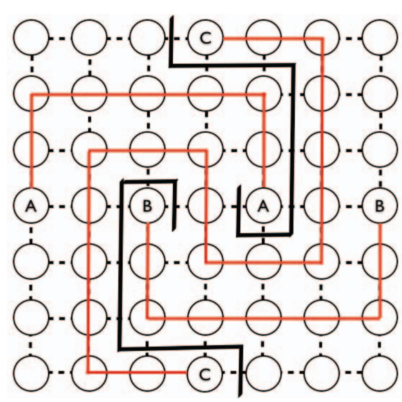

Fig. 3. An example of the difficult construction of walls

are also shown by the thick lines. In this case, it seems to be difficult to construct walls. As the authors in [6] mentioned, the wall construction has almost the same difficulty as the construction of the routing.

In this paper, we propose another fixed-length routing algorithm based on the color-coding algorithm, called *recoloring*. It is proved that recoloring outputs the feasible solution statistically, and it may consume much small runtime in most cases. We also confirm the efficiency of the proposed method empirically.

The rest of this paper is organized as follows: Section II gives a definition about the fixed-length routing problem and introduces the color-coding algorithm. In section III, we introduce recoloring based on the color-coding algorithm to solve the Multi-Path problem. We show the experimental results in section IV. In the last section, we conclude our research.

II. Preliminaries

In this section, we define the fixed-length routing problem and introduce the color-coding algorithm for the preparation of our discussion.

A. The Fixed-Length Routing Problem

The fixed-length routing problem in this paper is defined as follows: its input consists a grid graph $G = (V, E)$, a set of the terminal pair $N = \{n_i = (s_i, t_i)\}$ and the required length of each net l_i; its output is a family of the set of vertices which corresponds to the routing of the nets, where each net does not share any vertex with the other nets; the objective is to minimize the difference between the required length and the realized length. In this problem, the net length is counted by the number of edges passed through by the net.

In Fig.4, an example consisting of three nets is shown. In this figure, net n_1, n_2, and n_3 consist of $(1, 7, 8, 9)$,

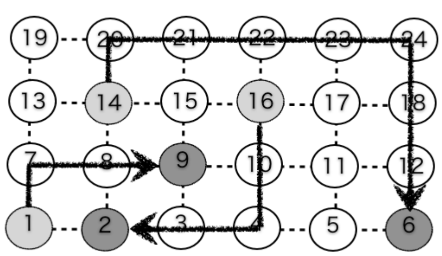

Fig. 4. An example of three nets, where $n_1 = (1, 9)$, $n_2 = (16, 2)$, $n_3 = (14, 6)$, $l_1 = 3$, $l_2 = 4$, and $l_3 = 8$

$(16, 10, 4, 3, 2)$, and $(14, 20, 21, 22, 23, 24, 18, 12, 6)$, respectively, where each net passes through the vertices exclusively and constructs a path. The lengths of n_1, n_2, and n_3 in Fig.4 are 3, 4, and 8, respectively. Thus, all nets satisfy their length requirements. This problem has been proved as $\mathcal{NP}$-hard [6, 7]. Thus, to obtain the optimized solution, the exhausted algorithm should be needed.

B. The Color-Coding Algorithm

In the remaining part of this section, we introduce the color-coding algorithm.

B.1. Definition and Example

For a given graph $G(V, E)$, let S and T be the terminals, where $S, T \in V$, and l is a positive integer. The l-length path N from S to T denotes as $N(S, T, l)$. For this problem, [8] has proposed an randomized algorithm called *color-coding*. The color-coding algorithm with c colors is shown in Fig.5.

step 1: Paint each vertex with a random color over c colors, where c should satisfy $c \geq L = l + 1$.

step 2: Search a path from S to T by the depth first search manner, where the path can pass through at most one vertex painted one color. If the path passes through k vertices with different colors, is is said to be *k-colorful*.

step 3: If a L-colorful path is found, remain the path and go to **step** 4. If a k-colorful ($k \neq L$) path is found and the difference between k and L is smaller than the best until the time, remain the path. If the number of trials is less than $\frac{L^L}{L!}$, go to **step** 1. Otherwise, go to **step** 4.

step 4: Recover the remaining path and output the path.

Fig. 5. Color-coding algorithm

To obtain the path $N(S, T, l)$, the color-coding algorithm searches L-colorful path from S to T. Note that it is necessary to remain not the vertex set but the color set to be passed. Figure 6 shows an example of path $N = (S, T, 4)$ with 5-color painting and 5-colorful path from S to T.

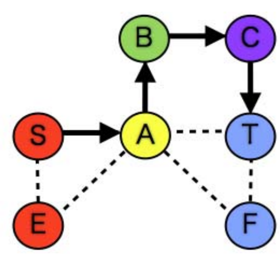

Fig. 6. An example of path $N = (S, T, 4)$

Figure 7 shows the depth-first search tree for the example shown in Fig. 6 from the color-coding algorithm shown in Fig.5. For the coloring in Fig.6, after visiting the vertex S, the traverse cannot break into the vertex E, since the colors of S and E are same. After the visiting A, the traverse cannot break into the vertex S, either. The traverse passes through B, C, and T. Then, the 5-colorful path between S and T is found. Only one vertex among the same colored vertices can be passed, thus the color-coding can reduce the explore space.

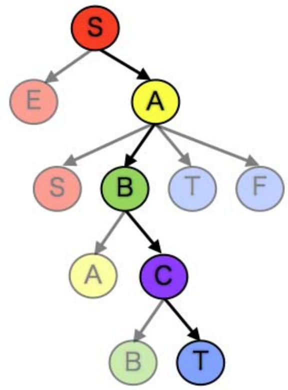

Fig. 7. A traverse tree of color-coding algorithm with depth-first exploration

B.2. Complexity of Color-Coding Algorithm

In this section, we discuss the complexity of the color-coding algorithm. To simplify the discussion, we assume that there is only one path for $N(S, T, l)$. For $k = l+1$, $N(S, T, l)$ is k-colorful path, if the color-coding algorithm finds the path. The number of the combinations of coloring on the path is k^k and the number of the combinations of the coloring of k-colorful path is $k!$. Thus, the probability to find the k-colorful path is $\frac{k!}{k^k}$. It is statistically guaranteed that one k-colorful path is obtained by at least $\frac{k^k}{k!}$ times painting of graph G.

Next, we discuss the complexity to obtain one k-colorful path for one k-coloring of G. For visiting each vertex, the variation of the color set to be passed should be considered. Since the number of colors is k, then the number of variations is $O(2^k)$. Furthermore, all vertices are visited in the worst case. Thus, the complexity to obtain one k-colorful path for one coloring of G is $O(n \cdot 2^k)$.

Therefore, to obtain one k-colorful path with k-coloring consumes $O\left(\frac{k^k}{k!} \cdot n \cdot 2^k\right)$. From Stirling's formula, $\frac{k^k}{k!} \approx e^k$, the complexity of color-coding algorithm is $O\left(e^k \cdot n \cdot 2^k\right)$.

III. The Fixed-Length Routing Method Based on the Color-Coding Algorithm

In this section, the proposed method is described.

A. Single-Path problem

The color-coding algorithm is proposed to solve the l-length routing problem. Thus, it is easy to apply it to the fixed-length single-path problem. To confirm the efficiency of the algorithm, we apply it to the fixed-length single-path problem on 15×15 grid graph. The terminals of the path exist with the next of the neighbor, shown in Fig.8,

The table I shows the results for the various fixed-length, which consists of the fastest time and the average time over 50 trials.

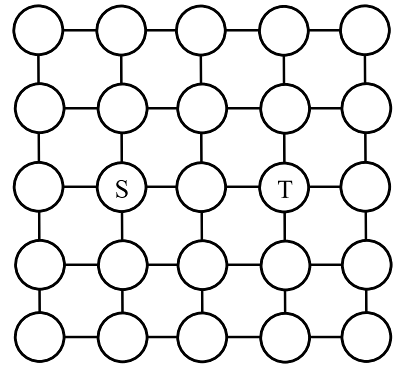

Fig. 8. An example of single-path

TABLE I
The runtime for the various fixed-length [ms]

Fixed-Length	Colors	Fastest	Average
10	11	0.04	0.06
20	21	3.33	5.05
30	31	95.43	168.36
40	41	677.33	9451.72
50	51	35632.59	71030.75

According to the result, the color-coding algorithm is efficient when the fixed-length is at most 40. While, if the fixed-length exceeds 40, the color-coding consumes much runtime, thus, it is not efficient.

As described above, the probability to find the k-colorful path is $\frac{k!}{k^k}$. Thus, for $k' > k$, the probability to find the k'-colorful path is $\frac{k'!}{k'^{k'}}$. The ratio between the probabilities of k-colorful path and k'-colorful path is

$$\frac{\frac{k'!}{k'^{k'}}}{\frac{k!}{k^k}} = \frac{k'! \cdot k^k}{k'^{k'} \cdot k!} = \frac{k'!}{k! k'^{(k'-k)}} \frac{k^k}{k'^k} < 1.$$

Thus, as the fixed-length becomes longer, the probability becomes smaller.

Next, the effect of the number of colors for a fixed-length routing is considered. For k-length routing, the number of colors is set to $k_{\mathrm{new}}(> k)$. In the case, the probability is

$$\prod_{i=0}^{k-1} \left(1 - \frac{\mathrm{i}}{k_{\mathrm{new}}}\right)$$

It means that the probability becomes larger if the number of colors becomes larger. While, the complexity to find the k-colorful path with k_{new}-coloring is $O(n \cdot 2^{k_{\mathrm{new}}})$. Thus, there is a trade-off for runtime.

The tableII and III show the runtime for the various colors with 40-length and 50-length, respectively. From the results, when the number of colors is about 1.2 or 1.3 times larger than the fixed-length, the running time becomes shorter. These results also show the existence of the trade-off.

TABLE II
The runtime for the various colors [ms] with 40-Length

Colors	Fastest	Average
41	677.33	9451.72
42	197.03	1948.74
44	41.56	265.94
46	2.41	73.47
48	0.36	46.20
50	0.26	35.62
52	1.40	5.83
54	0.17	12.11

TABLE III
The runtime for the various colors [ms] with 50-Length

Colors	Fastest	Average
51	35632.59	71030.75
55	46.27	466.47
58	36.97	311.78
60	0.08	90.82
62	1.04	30.00
65	1.40	12.88
67	5.42	115.64

B. *Multi-Path problem*

In [9], the color-coding algorithm is proposed to solve the balanced-path problem. A naive method to solve the fixed-length multi-path problem utilizes the method for the balanced-path problem. It transforms the fixed-length multi-path problem into the single-path problem. For the given nets, a net N^* is made by connecting each net sequentially, where the edges from T_i to S_{i+1} are added. Furthermore, the required length of N^* is $l^* = (\#nets) - 1 + \sum l_i$. Thus, finding the multi-path with their required length is just as finding k^*-colorful path, where $k^* = l^* + 1$.

However, this method could be too late to apply practical problems. For example, we applied this method to the problem shown in Fig. 3 with $l_A = 8$, $l_B = 8$, and $l_C = 18$. The program executed for 1 day with two hundred million colorings. Nevertheless, it does not output any feasible solution. The reason comes from too small probability described in the previous section. The probability to find the path is $\frac{k^*!}{k^{*k^*}}$, when the number of colors is just k^*. Thus, a heuristic algorithm to increase the probability is needed.

C. *Recoloring*

In order to increase the probability to find the path, we consider to reduce the number of colors. However, less color than the required length of one net leads the failure of routing.

We propose a method called *recoloring* shown in Fig.9.

In Fig.9, the distinctive colored vertices hold the routing for one net. Thus, the probability to find the routing may become large. In the worst case, when each vertex is painted by different color, we can obtain the optimum solution. The reason is that the case is equivalent to the exhaustive search. In order to improve the efficiency of this heuristic algorithm, HABCC algorithm in [10] may be utilized but it remains in our future works.

D. *An example of recoloring*

In this section, an example of recoloring is shown. We consider the 5×5 grid graph shown in Fig. 10, where the terminal sets of n_1 and n_2 are (L, N) and (G, T), respectively, and their requirement lengths are 4 and 7, respectively.

step 1: Set c to (the maximum required length of the net)+1.

step 2: Iterate the following steps until satisfying the stopping criterion:

- **step 2.1:** Paint each vertex with a random color over c colors except for the vertices colored by the distinctive colors.
- **step 2.2:** For n_i, search a path from S_i to T_i by the depth first search manner, where the path can pass through only one vertex painted the same color and it cannot share the vertex with other paths.
- **step 2.3:** If the routing of all nets with the predefined quality are found, then return the routing. Otherwise, paint the distinctive colors for the vertices on the n_i with the completion between S_i and T_i and go back 2.1.

Fig. 9. Recoloring

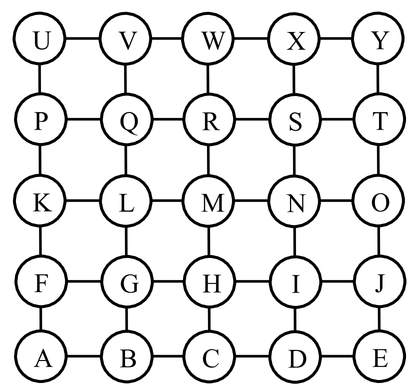

Fig. 10. 5×5 grid graph

In the first coloring, the number of colors is $8 = \max(4+1, 7+1)$. One coloring and the resultant path is shown Fig. 11. In the result, we can find the 2-length path for n_1 only. Thus,

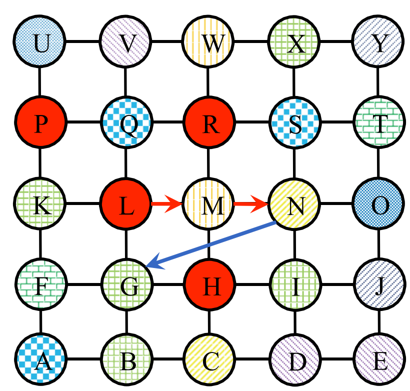

Fig. 11. First coloring and partial routing result

we paint the distinctive colors for L, M, and N.

In the second coloring, the number of colors is $11 = \max(4+$

1, 7 + 1) + 3. One coloring and the resultant path is shown Fig. 12. In the result, we can find the 4-length path for n_1.

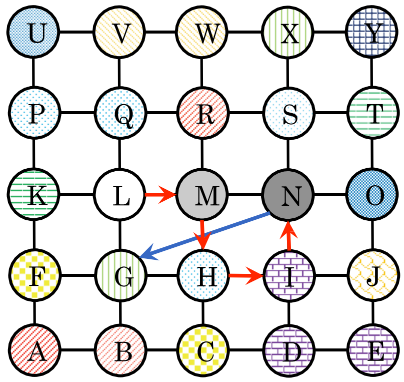

Fig. 12. Second coloring and partial routing result

The length matches the required length. But, we cannot find a path for n_2. Thus, we also paint the distinctive colors for five vertices on n_1.

In the third coloring, the number of colors is 13 = max(4 + 1, 7 + 1) + 5. One coloring and the resultant path is shown Fig. 13. In the result, we can find the 4-length and 7-length

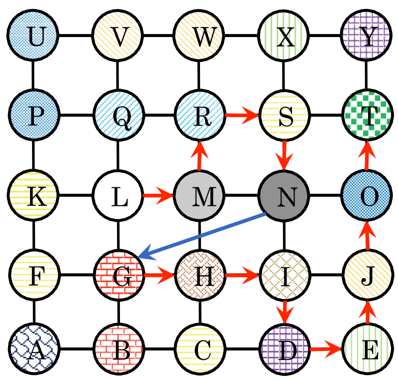

Fig. 13. Third coloring and routing result

paths for n_1 and n_2, respectively. They satisfy the required lengths, thus, they are output.

IV. Experimental Results

To confirm the proposed method efficiency, we implement the algorithm. The computation environment is Ubuntu 12.04.1, gcc 4.6.3, on Intel(R) Core(TM) i7 CPU 920 @ 2.67GHz with 6GB memory.

At first, we execute the single-path problem. The experimental results are shown in Table IV, where the column FL, PM, and ES correspond to the fixed-length, the runtime of the proposed method, and the runtime of the exhaustive search, respectively. For each data, the initial value of the number of colors is set to FL + 1.

All case can be obtained the optimum solution. Compared with the exhaustive search, the proposed method is much

TABLE IV
The runtime for Single-Path Problem [ms]

Size	FL	PM	ES
10*10	11	0.002	0.003
10*10	15	0.003	0.06
15*15	70	0.31	372.59
20*20	100	6.90	5358.37

faster. We confirm recoloring is efficient to the fast computation.

Next, we apply recoloring to the multi-net problems. The experimental results are shown in Table V, where PM and ES also correspond to the runtime of the proposed method and the runtime of the exhaustive search, respectively.

TABLE V
The runtime for Multi-Path [s]

data-id	PM	ES
1	0.028	19.59
2	2.56	22.09
3	8.47	102.12
4	0.01	0.03

Furthermore, Fig.14, Fig.15, Fig.16, and Fig.17 show routing results for data-1, data-2, data-3, and data-4, respectively. In the figures, the circles and the black colored areas correspond to the terminals and the obstacles, respectively.

From the experimental results, the proposed method is efficient. Especially, the terminals in data-3 and data-4 do not satisfy the river routing condition. As the previous section described, CAFE router needs the "wall" constructions to satisfy the river routing constraint. However, paper [6] dose not propose any method for "wall" construction. Thus, CAFE router cannot be executed for data-3 and data-4, directly. On the other hand, the proposed method can apply all data. Therefore, we confirm the proposed method is efficient.

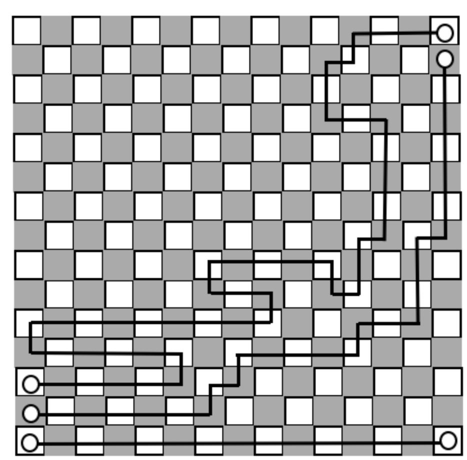

Fig. 14. A result for data-1: $(l_1, l_2, l_3) = (14, 26, 50)$

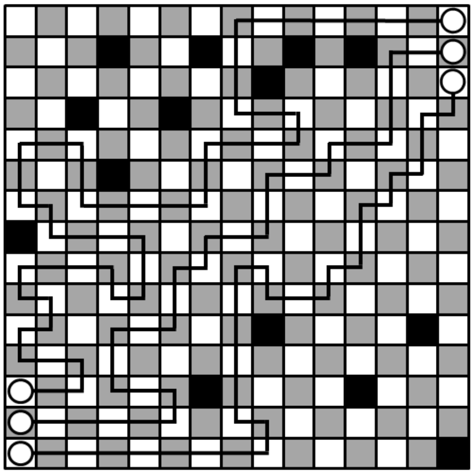

Fig. 15. A result for data-2: $(l_1, l_2, l_3) = (30, 30, 50)$

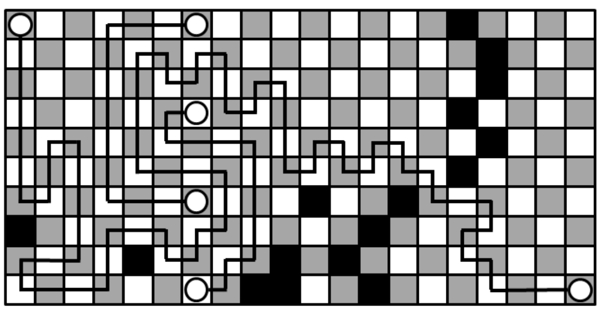

Fig. 16. A result for data-3: $(l_1, l_2, l_3) = (12, 12, 70)$

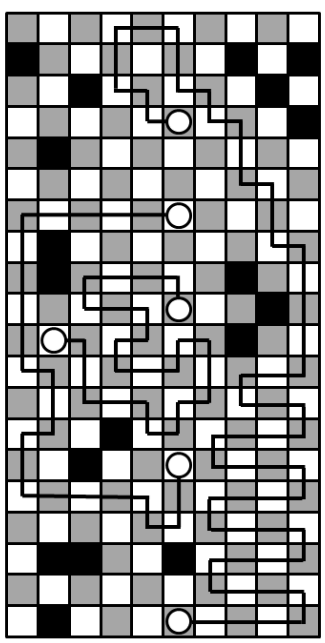

Fig. 17. A result for data-4: $(l_1, l_2, l_3) = (24, 25, 56)$

V. Conclusion

In this paper, we propose a fixed-length routing method based on the Color-Coding algorithm. According the results, the proposed approach is able to find Multi-Path in fixed-length without setting "walls", and it is much faster than the exhaustive search. Thus, we confirm the proposed method is efficient.

In the future works, we can enumerate

- application of the proposed method to the large data,
- more efficient search scheme, and
- enhancement to the multi-layer routings.

References

[1] R. Y. Pinter, "On Routing Two-Point Nets Across a Channel," *in DAC,* pp. 894-902,1892.

[2] C. P. Hsu, "Channel River Routing Algorithm," *in DAC,* pp. 578-583,1983.

[3] M.M Ozdal and M.D.F. Wong, "Algorithmic study of single-layer bus routing for high-speed boards," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems,* 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. on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 25, no. 12, pp: 2784-2749,2006.

[5] T.Yan and M.D.F. Wong, "BSG-Router: A length-matching router for general topology," in proc. of *International Conference on Computer-Aided Design,* pp. 499-505, 2008.

[6] Y.Kohira and A.Takahashi, "CAFE router:A fast connectivity aware multiple nets routing algorithm for routing grid with obstacles," in proc. of *Asia and South Pacific Design Automation Conference,* pp.281-92,2000.

[7] Y.Kohira, S. Suehiro and A.Takahashi, "A fast Longer Path Algorithm for Routing Grid with Obstacles using Biconnectivity based Length Upper Bound," in proc. of *ASPDAC,* pp.600-605,2009.

[8] N. Alon, R. Yuster, and U. Zwick, "Color Coding," *Journal of the ACM,* 1995,42(4): 844-856.

[9] P. Cappanera and M. G. Scutella, "Color-coding Algorithms to the Balanced Path Problem: Computation Issues," *INFORMS J. Comput.,* 2011,23(3): 446-459.

[10] J.-X. Wang, Z.-B. Yand, Y.-L. Liu, and J.-E. Chen, "An Improved Color Coding Algorithm Based on Hybrid Architecture," *Chinese Journal of Computers,* 2010,33(6): 1025-1031.