A Novel Detailed Routing Algorithm with Exact Matching Constraint for Analog and Mixed Signal Circuits[*]

Qiang Gao, Hailong Yao, Qiang Zhou and Yici Cai
Department of Computer Science and Technology, Tsinghua Univeristy, P. R. China
gaoq04@gmail.com, {hailongyao, zhouqiang, caiyc}@mail.tsinghua.edu.cn

ABSTRACT

In analog and mixed signal designs, exact matching requirement is critical for correct functionality of analog devices. However, due to the excessive complexity, it is difficult to consider exact matching constraint in detailed routing stage. This paper presents a novel gridless detailed routing algorithm, which efficiently obtains the optimized detailed routing solutions for a given set of nets with exact matching constraints. The gridless routing algorithm is based on an efficient non-uniform grid model, which enables the obstacles avoidance. To verify the effectiveness of the gridless routing algorithm, a grid routing algorithm and a modified exact matching routing algorithm from [5] are also implemented. Experimental results show significant improvements of the proposed gridless routing algorithm over the other two algorithms in both QOR and runtime.

Keywords

analog and mixed signal design, exact matching routing, detailed routing, gridless routing

1. INTRODUCTION

As VLSI technology advances, the functions of ICs are getting diversified and the integration of digital and analog technologies is becoming increasingly important. The past half-century has seen remarkable achievements in both theory study and tool development in digital design automation. However, analog signals are more sensitive than digital signals, and hence analog design automation has not made the same headway due to the complicated electrical effects and constraints, such as geometric constraint, variable line widths, crosstalk, antenna effect, etc. As a result, analog design automation is becoming the bottleneck of the whole design process of modern SOC and ASIC designs with analog and mixed signal components. Nowadays, automatic analog design, especially analog routing is still a challenging topic which attracts more and more attentions [1][7][9].

The matching requirement for nets is a prevalent geometric constraint in analog and mixed signal designs, even in digital designs. For example, in analog circuit amplifier and the high-performance clock trees in digital ciruits, delay mismatch between wires needs to be reduced to enhance the performance. Functionalities of these circuits depend directly on the matching constraint. In previous research works [6][8][10], the matching constraint is converted to wire-

[*]This work is supported in part by the National Natural Science Foundation of China (60976035) and in part by Specialized Research Fund for the Doctoral Program of Higher Education:SRFDP-200800030026.

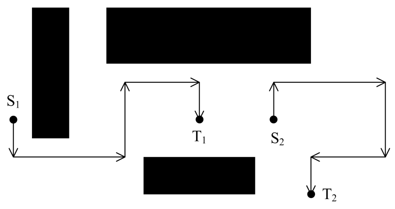

Figure 1: An example for which directly extending algorithm in [5] is not applicable.

length matching. However, length matching is not enough to ensure the identical electrical properties of analog circuits. In multilayer IC designs, wires and vias on different layers have different electrical parameters, which influence the properties of analog circuits. In [5], the SPICE simulations for three pairs of nets are conducted to illustrate the limitation of length matching constraint. Though the wirelengths of all the nets are 250um, the delay mismatch between two nets with the exact matching constraint is 0ps, while the delay mismatch of the other two pairs with the length matching constraint is about 0.6ps. Therefore, in order to obtain high electrical performance, exact matching constraint needs to be enforced, where segments of routes are of the same lengths and on the same layers, and the via numbers are also the same for each cut layer.

Carragher et al. propose a linear programming based algorithm for computing the optimal Steiner point positions to equalize wirelengths and to achieve zero skew [2]. In [6], a Lagrangian relaxation based routing algorithm is proposed to route nets within tight minimum and maximum length bounds. Yan et al. propose a routing algorithm based on Bounded-Sliceline Grid (BSG) to solve length matching problem [10]. The above-mentioned works consider the length matching constraint, and the nets are routed on a single layer. Therefore, the proposed methods may not be applicable in modern multilayer IC design. Algorithms for the length-compensation of in multilayer bus routing are presented in [8], where short wires have to snake for equal wirelength, which introduces more vias and hence greater delay mismatch. In [4], Du et al. propose a routing algorithm with symmetric constraint, which is a stricter constraint than the exact matching constraint.

Recently, Ozdal et al. propose a dynamic-programming algorithm based on a mathematical model for exact matching routing in [5], which is applied during global routing stage to plan the routing topologies. The algorithm predetermines the value of matching configuration matrix (defined in [5]) and does not obtain all feasible solutions for each possible configuration matrix. As a result, the solution space for exact matching problem is reduced (i.e., some optimal solutions are lost) and direct extension of this algorithm (e.g., removing edges to handle obstacles) is not suitable for detailed routing. Another constraint from the configuration matrix is that the number of bends cannot exceed $2 \times |N| - 1$, where $|N|$ is the total number of nets with the exact matching constraints. For example, for the two nets in Figure 1 with exact matching constraints, if more than 3 bends are needed to construct the exact matching routes, simple extension of the algorithm in [5] cannot obtain the correct routing solution. Our exact matching routing algorithm obtains the routing solution as shown in Figure 1.

Note that the sizes of current analog and mixed signal circuits are often small enough for direct detailed routing without global routing, which strengthens the motivation for research on detailed exact matching routing algorithms. In this paper, a novel detailed gridless routing algorithm is proposed to efficiently obtain the exact matching routing solutions for a given set of two-pin nets. Our algorithm guarantees to find the routing solution as long as there exists one, with the exact matching constraints observed and routing obstacles avoided. To the best of our knowledge, this is the first detailed routing algorithm that considers the exact matching constraint and obstacle avoidance simultaneously. The major contributions of this paper are as follows.

- We classify bends/vias of nets into three categories based on their purposes: (1) heading to routing targets, (2) avoiding obstacles, and (3) matching nets exactly. Based on the three categories, we propose the routing strategy to obtain the exacting matching routing solutions.
- The routing area is partitioned according to the coordinates of all potential bends/vias. Then a non-uniform grid model is adopted to improve the efficiency of our gridless routing algorithm.
- We propose an innovative detailed routing flow to observe the exact matching constraint, where nets are expanded simultaneously to guarantee the segments and vias of all routes to be identical. An improved A*-search algorithm is adopted to consider the matching error during the expanding stage, which further reduces the runtime.

The rest of the paper is organized as follows. Section II introduces the exact matching routing problem. In Section III, we analyze the conditions where bends occur and propose our gridless routing algorithm based on A*-search. Experimental results are presented and discussed in Section IV. Finally, the paper is concluded with future research directions in Section V.

2. PRELIMINARIES

The exact matching routing problem can be stated as follows: given a multilayer routing area where each layer has a preferred routing direction[1], a set of rectangular obstacles, and a set of nets along with exact matching constraints, find the optimized detailed routing solutions for the nets with the exact matching constraints observed.

Definition 1 Exact Matching Constraint[2]**:**

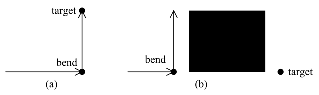

Figure 2: Different types of bends according to [3]. (a) bends for expanding towards targets; and (b) bends for avoiding obstacles.

1. All the nets have the same segment count.
2. The lengths of the respective segments are equal.
3. The layer assignment of the respective segments is equal.

The objectives of exact matching routing algorithm are as follows:

1. Finish connecting source and target of each net, while avoiding all obstacles.
2. Guarantee all the given nets be exactly matched.
3. Minimize the total cost (including wirelength and vias).

3. EXACT MATCHING ROUTING

The traditional detailed routing algorithm usually partitions the layout into uniform grids, and then searches the route on the grids. However, grid routing model is not very efficient and may waste the routing resource. Exact matching routing problem is more complex than routing a single net; what is more, routing the whole nets from sources to targets can optimize the wirelength and the number of vias to obtain much better electrical properties than routing nets through two stages (global routing and detailed routing), so grid model is not suitable for exact matching routing.

Enlightened by [3], the whole routing region can be partitioned into non-uniform grids according to the bends' locations. Applying this non-uniform model can accelerate the maze expanding process and make full use of the routing resource. Based on this idea, we propose a novel gridless detailed routing algorithm to handle the exact matching constraint. The main flow of our gridless detailed routing algorithm can be summarized as follows:

1. Determine the coordinates of all the potential bends (in Section 3.1).
2. Partition the layout into non-uniform grids (in Section 3.2).
3. Route the given set of nets to satisfy exact matching constraint (in Section 3.3∼3.5).
4. Do congestion avoiding if needed (in Section 3.6).

3.1 Bend Analyses

According to [3], there are two types of bends. One is for heading to targets and the other is for obstacle avoidance. Examples are illustrated in Figure 2. It is easy to compute the coordinates of the two kinds of bends: they are aligned with either the routing terminals or the boundaries of the obstacles.

However, during exact matching routing process, nets have to snake to make routes exactly matched. So routes may bend even if there are no obstacles, as shown in Figure 3. The problem of determining the coordinates of the bends for the exact matching constraint can be simplified to as computing the length of each horizontal or vertical segment, which is formally stated in the rest of this subsection.

[1]In this paper, we consider the exact matching routing problem in the context of restricted routing mode, i.e., each routing layer has a single routing direction, either horizontal or vertical.

[2]More detailed description of the exact matching constraint can be found in [5].

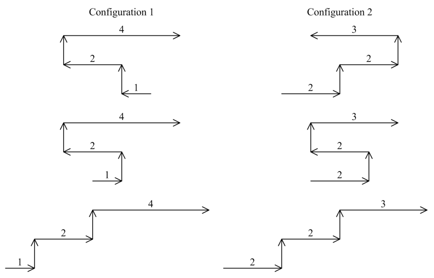

Figure 3: A new type of bend for satisfying the exact matching constraint.

Let $D_H[i]$ ($D_V[i]$) represent net i's horizontal (vertical) distance from source to target($1 \leq i \leq N$, N is the number of nets along with exact matching constraint). Each route has k_H horizontal unit segments and k_V vertical unit segments, and L_H (L_V) represents the vector of horizontal (vertical) lengths of these unit segments. $C_H[n, i]$ ($C_V[n, i]$) is the coefficient of $L_H[n]$ ($L_V[n]$), whose value is 1 or -1 (n represents the order of each unit segment).

The problem of computing coordinates of potential bends for the exact matching constraint can be solved by computing L_H and L_V subject to the following constraints.

$$\sum_{n=1}^{k_H} C_H[n,i] \cdot L_H[n] = D_H[i] \qquad (1)$$

$$\sum_{n=1}^{k_V} C_V[n,i] \cdot L_V[n] = D_V[i]$$

In [5], a mathematical method is presented to obtain one feasible solution of L_H and L_V for a predetermined configuration matrix. However, for a set of given nets, the configuration matrix (C_m) is not unique and there are many feasible solutions for one C_m. Figure 3 shows two solutions of L_H for an example with 3 nets (Configuration 1: $L_H = [1, 2, 4]$ and Configuration 2: $L_H = [2, 2, 3]$). Applying the mathematical method as in [5] cannot determine all the potential coordinates of the bends for the exact matching constraint.

To handle this problem, a recursive algorithm (Algorithm 1) is developed to improve the mathematical model and obtain all the possible L_H's and L_V's. The algorithm is provided for better understanding the algorithm. In the implementation, a for-loop based algorithm in used instead of the recursive algorithm for better runtime performance. The coordinates of the bends keeping nets exactly matched can be obtained by permuting the order of $L_H(L_V)$ and $C_H(C_V)$.

As shown in Figure 3, routes are alternate between horizontal and vertical segments. If there is 0 in L_H or L_V, segments may not be enough to construct the routes. According to [5], long segments are split into short ones to add extra segments. For example, if $L_H = [0, 0, 3]$, we can use $L_H = [1, 1, 1]$ instead.

3.2 Non-Uniform Grid Model

Since the locations of all the potential bends have been determined, we can define the non-uniform grid graph for exact matching routing problem. Obstacles are expanded according to wire width/spacing and via width/spacing to keep the route satisfying design rules. Then, the routing region is partitioned according to the boundaries of expanded obstacles, source and target points and bends for exact matching constraint.

Assume there are 2 nets and both of them turn inside non-uniform grids. Since the coordinates of target points are on the grid edges, the two nets would turn again at somewhere. According to the relative positions of segments, there are two different kinds, as shown in Figure 4.

In Figure 4 (a), the relative positions of segments are the same. The middle segment of net 1 could move left to the grid edge, and the middle segment of net 2 can move right with the same distance. Doing this movement does not violate the exact matching constraint.

In Figure 4 (b), the relative positions of segments are different. The middle segments of the two nets cannot move, and these bends are for the exact matching constraint. Thus, the coordinates of these bends should be calculated by Algorithm 1.

In sum, when bends happen during expanding, the bend of one net should be on the non-uniform grid. Figure 5 illustrates an example of non-uniform grid graph along with the exact matching routing result.

3.3 Simultaneous Routing

Previous routing algorithms for the length matching constraint typically route nets first and then snake the wires after routing to increase the wirelengths of short nets [6][8]. But when detailed routing finishes, most parts of the routes are determined and there are limited routing resources for adjusting the routes. So it is difficult for these algorithms to keep nets exactly matched through snaking after detailed routing. We propose an exact matching routing algorithm,

Algorithm 1 CalculateL() -Recursive Algorithm for Calculating $L_H(L_V)$

```
Input: $D_H$, $L_H$, $size$
 1: Randomly choose $D_H[p]$ and $D_H[q]$
 2: Pop $D_H[p]$ and $D_H[q]$ from $D_H$
 3: $value = |D_H[p] + D_H[q]| / 2$
 4: $diff = |D_H[p] - D_H[q]| / 2$
 5: if $size == 2$ then
 6:     Push $value$ and $diff$ into $L_H$
 7:     return
 8: else
 9:     $n = D_H.size$
10:     for $index = 0; index < 2^n; index++$ do
11:         $pos = 1$
12:         for $i = 0; i < n; i++$ do
13:             $val = index \& pos >> i$
14:             $D_H[q] += (val * 2 - 1) * diff$
15:             $pos *= 2$
16:         end for
17:         Push $value$ into $D_H$
18:         Push $diff$ into $L_H$
19:         CalcualteL($D_H$, $L_H$, $size - 1$);
20:     end for
21: end if
```

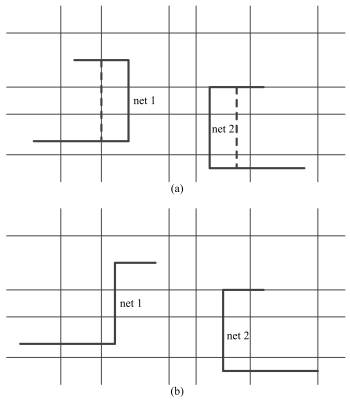

Figure 4: Proof of Non-uniform grid graph.

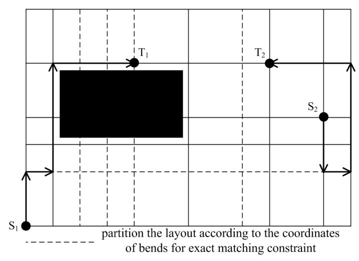
T1
T2
S2
S1
partition the layout according to the coordinates
of bends for exact matching constraint

Figure 5: Non-uniform grid graph.

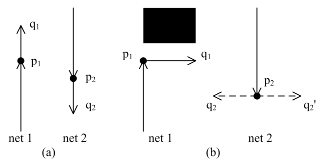

Figure 6: Expanding process.

where the exact matching constraint is considered during the routing process. In our algorithm, exact matching nets are expanded simultaneously, as shown in Figure 6. The expanding process is stated as follows.

1. One net is chosen as *chief* net[3].
2. Other nets are checked on whether they can be expanded. Those nets should be expanded with the same or opposite orientation and with the same length due to the exact matching constraint.
3. If such expanding step is available, new states are generated and the expanding process will continue; otherwise, such expanding step would be ignored.

Figure 6 illustrates the process of expanding two nets at the same time. Assume the current state is $[p_1, p_2]$, and net 1 is set as the chief net. In Figure 6(a), net 1 goes forward at point p_1, and net 2 would maintain its previous expanding orientation. Then only one state $[q_1, q_2]$ is generated. In Figure 6(b), net 1 goes from a vertical layer to a horizontal layer and turns right. Then net 2 expands to the same horizontal layer. But net 2 can turn both left and right. Then two different states $[q_1, q_2]$ and $[q_1, q_2']$ are generated.

3.4 Improved A*-Search Algorithm

Since nets propagate simultaneously, an expanding state should contain the expanding points' coordinates of all the given nets. In our data structure, each state contains a vector of points to record the positions of expanding points. Besides, to ameliorate the heuristic conditions of the A*-search algorithm, more variables are needed, as listed in Table 1. Besides, some variables are needed in A*-search as shown in Table 2.

The values of mat_err, $costH$, $costG$ and $costF$ are computed as follows.

$$\begin{aligned} mat\_err &= \sum_{n=1}^{N}(max\_d_H - d_H^n) \cdot min\_cost_H \\ &\quad + (max\_d_V - d_V^n) \cdot min\_cost_V \\ costH &= max\_d_H \cdot mCost_H + max\_d_V \cdot mCost_V \qquad (2) \\ &\quad + |cur\_s.layer - tar\_s.layer| \cdot cost_{via} \\ costG &= pre\_s.costG + \Delta g \\ costF &= costG + costH \end{aligned}$$

[3]Section 3.5 explain the chief net selection in detail.

Table 1: Member variables of an expanding state.

$pVec$	a vector recording the coordinates of expanding points
$layer$	the layer assignment of the expanding points
mat_err	the matching error estimate of the current state
$costG$	the real cost from the source to the current state
$costH$	the estimate of cost from current state to the target state
$costF$	total cost for A*-search

Table 2: Variables used in A* search.

N	a set of nets which are exactly matched
n	a net of $N (n \in N)$
d_H^n	the horizontal distances from net n's expanding point to n's target point
d_V^n	the vertical distances from net n's expanding point to n's target point
max_d_H	the maximum d_H^n in N
max_d_V	the maximum d_V^n in N
$mCost_H$	minimum expanding cost among all horizontal layers
$mCost_V$	minimum expanding cost among all vertical layers
$cost_{via}$	the cost of one via
Δg	the cost of current expanding step
cur_s	the current state
pre_s	the state of the previous step
tar_s	the target state

Based on the expanding process and data structure discussed above, the improved A* algorithm for nets with the exact matching constraint is shown in Algorithm 2.

Algorithm 2 A*-search-based Routing Algorithm for Exact Matching constraint

```
Input: $source\_state$, $target\_state$, $route\_area$
 1: $openlist = source\_state$
 2: while $openlist$ is not empty do
 3:   $current$ = the first element of $openlist$
 4:   Remove $current$ from $openlist$
 5:   if $current == target\_state$ then
 6:     Find the paths
 7:     return true
 8:   else
 9:     Obtain expanding orientations and lengths for each net
10:     Determine chief net
11:     Generate new states according to the expanding orientations and lengths of chief net
12:     Calculate $mat\_err$, $costH$, $costG$, $costF$ for the current state
13:     Insert new states into $openlist$ to keep $openlist$ sorted according to $costF$ and $mat\_err$ in ascending order
14:   end if
15: end while
16: return false
```

In A*-search, keeping *openlist* sorted is quite a critical step to guarantee the correctness and efficiency of the routing algorithm. Because states with the same *costF* are sorted by *mat_err*, our algorithms always expand the state with smaller matching error. This cost function can remarkably reduce the searching space.

3.5 Details of the Routing Algorithm

To keep routes exactly matched, the expanding lengths for all nets of each expanding step should be identical. However, grids are non-uniform, and the lengths of grid edges are variable. So the expanding length must be determined first during expanding process. Moreover, since in most cases the expanding length is not equal to the length of grid edges, not all segments of routes are on the grid edges, and not all bends occur at grid nodes as well. Therefore, an expanding point may be at a grid node, on a grid edge and inside a

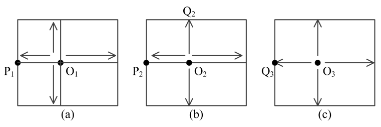

Figure 7: Expand from expanding point with no constraints: (a) at a grid node; (b) on a grid edge; and (c) inside a grid.

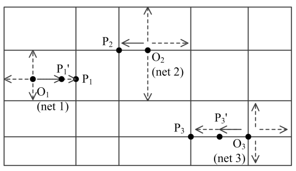

Figure 8: The process of setting chief net.

grid, as shown in Figure 7. This increases the difficulties of fixing the expanding length.

The rules of propagating from expanding point to the next one is stated as follows:

1. If the expanding point is at a grid node, it can expand to its neighboring grid nodes (from O_1 to P_1 in Figure 7(a)).

2. If the expanding point is on a grid edge, it can expand to the grid nodes on the same edge (from O_2 to P_2 in Figure 7(b)) or to points on the opposite grid edges across grids (from O_2 to Q_2 in Figure 7(b)).

3. If the expanding point is inside a grid, it can expand to points on the grid edges (from O_3 to Q_3 in Figure 7(c)).

In our gridless routing algorithm, the process of determining expanding length is performed in four steps:

1. Expand each net without any constraints and gain the expanding length of each net, as shown in Figure 8.

2. Choose an expanding direction for each net, which should be all in horizontal or vertical direction. As shown in Figure 8, net 1 is expanded from O_1 to P_1, and net 2 is expanded from O_2 to P_2, and net 3 is expanded from O_3 to P_3.

3. Choose the minimum expanding length as the expanding length of current expanding process. Set the net with this expanding length as chief net. As shown in Figure 8, since net 2 has the minimum original expanding length, it is set as chief net and its original expanding length is set as the expanding length of current expanding process.

4. Use this length to expand other nets and generates new states according to Section. As shown in Figure 8, finally, net 1 is expanded from O_1 to $P_1^{'}$, and net 3 is expanded from O_3 to $P_3^{'}$.

In analog signal designs routing resource is typically abundant; besides, the nets with exact matching requirements are critical nets, and they are of higher priority and routed before other nets. So obstacles such as pre-routes are few, and our exact matching routing algorithm can significantly reduce the count of grid nodes and runs efficiently.

3.6 Self-loops and Congestion Avoiding

Self-loops may occur during detailed routing to increase the length of some nets. To avoid this situation, every expanding state has been checked carefully whether expanding points are on the route from source points to current points. If self-loops happens, the expanding state would not be inserted into *openlist*.

Since the given set of nets with exact matching constraint is expanded simultaneously, and it is hard to check whether a point is on the routes of other nets being routed, sometimes nets may cross each other. Similar to [5], nets are ripped-up and rerouted iteratively and the cost of congestion region increases if congestion happens. In analog or mixed signal ICs, routing resources are sufficient, so congestion is eliminated after several iterations.

4. EXPERIMENTAL RESULTS.

The gridless detailed routing algorithm is implemented in C++ and run on Linux server with Intel(R) Xeon(TM) 3.00GHz CPU. The algorithm is tested by public benchmarks from ISPD'07 Global Routing Contest for to comparison with the algorithm proposed in [5]. In mixed signal design, analog and digital nets are routed on the same layout, so it is realistic to use the ISPD'07 benchmarks to evaluate our algorithms. We have also tested our algorithm on some industrial benchmarks. But the results cannot be made public due to IP issues.

Since ISPD benchmarks are designed for global routing stage, there are no obstacles. To make them applicable to detailed routing, the benchmarks are modified by the following strategies in order to add obstacles and model detailed routing: (1) the capacity of some GR grids is set to be 0, and nets cannot be routed in them; and (2) some nets are pre-routed and regarded as obstacles during exact matching routing. Since nets with exact matching constraint are critical nets and have to be detailed routed before other non-critical nets, obstacles are few when these nets are routed. In each benchmark, 10% GR grids and 0.2% pre-routed nets are randomly chosen as obstacles. For each benchmark we randomly choose more than 15 groups of 2 and 3 nets with Manhattan distance between 100 and 200 respectively to enforce exact matching constraint on. All these nets are routed on layers between Metal3 and Metal6. The public parameters given for ISPD'08 and ISPD'09 are used to evaluate delay mismatch through Spice.

For comparison, the algorithm proposed in [5] is implemented and tested, which is the most relevant work for exact matching constraint in the literature. Since the algorithm predetermines the value of configuration matrix and only obtain one feasible solution, it cannot be applied in detailed routing stage. We modify the algorithm to try all feasible solutions to avoid obstacles and find the routing result with minimum cost. Bus as stated in the introduction, simple extension of [5] still cannot find correct solutions for all the cases. We also implement a grid detailed routing version to verify the correctness of our gridless algorithm.

Although our benchmarks contain only a few obstacles, some groups of nets cannot be exactly matched due to the random obstacles and the complexity of exact matching constraint. According to Theorem **??**, both our grid and gridless routing algorithms guarantee to find the exact matching solution if there exists one. Therefore, those randomly generated testcases, which cannot be successfully routed, are regarded as non-routable. We have thoroughly checked those testcases to double confirm the conclusion. So we count the information of successful groups.

Figure 9 shows detailed routing results from adaptec4. S_1T_1 and S_2T_2 are two nets which should be exactly matched.

Table 3: Experimental Result of Detailed Routing

benchmark	#nets in per group	length aver.	grid algorithm		gridless algorithm				alg. in [5]
			delay mismatch (ps)	runtime (s)	delay mismatch (ps)	runtime (s)	#failed groups	speedups	#failed groups
adaptec1	2	178	0	144.54	0	3.71	1	39	9
	3	272	0	752.79	0	62.48	2	12	10
adaptec2	2	236	0	106.89	0	4.05	1	26	9
	3	234	0	679.62	0	65	3	10	9
adaptec3	2	213	0	224.15	0	7.73	2	30	8
	3	259	0	703.02	0	79	2	9	11
adaptec4	2	214	0	242.28	0	13.44	1	18	8
	3	289	0	812.31	0	72.01	4	11	9
adaptec5	2	221	0	173.68	0	7.41	2	23	8
	3	288	0	727.68	0	69.53	3	10	7
newblue1	2	186	0	165.38	0	8.97	3	19	9
	3	254	0	918.8	0	77.8	4	12	10
newblue2	2	198	0	296.55	0	12.38	2	24	7
	3	264	0	814.43	0	69.96	3	12	8
newblue3	2	225	0	167.48	0	7.38	1	23	9
	3	249	0	849.12	0	76.3	3	12	9
average	2	209	0	227.62	0	8.13	/	28	/
	3	264	0	782.22	0	71.51	/	11	/

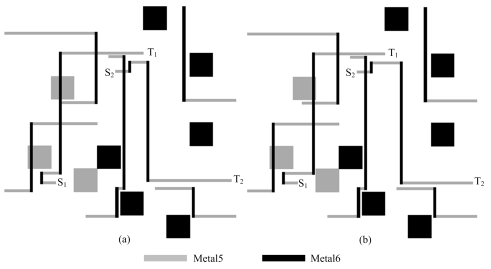

Figure 9: Detailed routing results: (a) grid algorithm; (b) gridless algorithm.

Both grid algorithm and gridless algorithm successfully find the routes with the exact matching requirement and minimum cost. The routing results are slightly different because the chief net changes during the routing process of gridless algorithm. By contrary, the modified algorithm fails in searching the correct routes.

Table 3 shows average wirelength, mismatch, total runtime of the successful groups and the failed groups. Our gridless routing algorithm can find routes satisfying exact matching constraint in most of benchmarks, while the modified algorithm from [5] fails in much more benchmarks because a lot of bends are needed to construct routes, i.e. more than 3 bends for the 2-net groups and more than 5 bends for the 3-net groups. We also verify that the groups of nets which cannot be routed by our gridless algorithm also fail by modified algorithm in [5]. Furthermore, gridless routing algorithm is much faster than grid routing algorithm (i.e., around 10X speedup), which proves the effectiveness of the non-uniform grid model.

Another important point to note is that matching 3 nets exactly takes much more runtime than matching 2 nets. We also observe that keeping 4 or more nets exactly matched often takes dozens of minutes. This is because: (1) keeping more nets matching needs much longer routes; and (2) for N nets, each expanding step will generate 2^N new states, and the expanding states increase dramatically with N. Therefore, the proposed algorithm is not recommended to route 4 or more nets simultaneously, though according to our experience, matching 2 nets is the most common exact matching routing requirement.

5. CONCLUSIONS

In analog and mixed signal designs, the exact matching constraint should be enforced for certain analog devices on to guarantee optimal electrical properties. In this paper, we present the first gridless detailed routing algorithm with the exact matching constraint, which guarantees to obtain the routing solution whenever there exists one. In current analog and mixed signal circuits, the size of the routing area with the exact matching constraint is often small enough to directly apply detailed routing. Thus, detailed exact matching routing algorithms prove to be of practical importance. Experimental results are promising and show that our algorithm is effective and efficient. In the future, we test our detailed routing algorithm on more industrial benchmarks. We will extend our algorithm to consider shielding and other geometric constraints. We will also investigate the bounded-error matching routing beyond the exact matching routing.

6. ACKNOWLEDGMENTS

We thank Prof. Andrew Kahng from UC San Diego for insightful advice and useful comments.

7. REFERENCES

[1] 2009 ITRS, http://www.itrs.net/.

[2] R. J. Carragher, C.-K. Cheng, and M. Fujita, "An efficient algorithm for the net matching problem", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 1993, pp 640-644.

[3] J. Cong, J. Fang, and K.-Y. Khoo,"DUNE: a multi-layer gridless routing system with wire plannin", *Proc. International Symposium on Physical Design*, 2000, pp 12-18.

[4] C. Du, Y. Cai, X. Hong, and Q. Zhou,"A shortest-path-search algorithm with symmetric constraints for analog circuit routin", *Proc. International Conference on ASIC*, 2005, pp 844-847.

[5] M. M. Ozdal and R. F. Hentschke, "Exact route matching algorithms for analog and mixed signal integrated circuit", *Proc. International Conference on Computer-Aided Design*, 2009, pp 231-238.

[6] 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, 2006, pp 2784-2794.

[7] R. A. Rutenbar, "Design automation for analog: the next generation of tool challenges", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2006, pp 458-460.

[8] M. A. Smith, L. A. Schreiner, E. Barke, and V. M. zu Bexten, "Algorithms for automatic length compensation of busses in analog integrated circuits", *Proc. International Symposium on Physical Design*, 2007, pp 159-166.

[9] N. Weste and D. Harris, "CMOS VLSI Design: A Circuits and Systems Perspective", Pearson Educatoin Asia Limited and China Machine Press, 2005.

[10] T. Yan and M. D. F, "BSG-Route: a length-matching router for general topology", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp 499-505.