Military Technical College
Kobry Elkobbah,
Cairo, Egypt

5th International Conference
on Electrical Engineering
ICEENG 2006

Novel routing algorithm using a bi-directional bubble sort for non-Manhattan channel routing.

Khaled A. Shehata, Amr M. Bauymi, Waleed Abd El-Hameed,
Tarif El-Shafiey.

Abstract

A non-Manhattan channel router uses fewer routing tracks than a Manhattan one. Many optimizations exist for the non-Manhattan channel routing to minimize the number of vias as well as the crosstalk between the layers [7, 8].

In this paper, we propose a direct implementation of an optimal three-layer bubble-sort-based non-Manhattan channel routing algorithm. This direct implementation minimizes the time complexity of the three-layer routing problem. We also implement a five-layer technique to reduce the channel height of our three-layer algorithm without affecting its time complexity.

1. Introduction

The reduction of the required chip area is an important goal in the automatic layout generation of VLSI circuits. The further reduction in width for the standard cell layout design is important. Most of the channel routing (CR) techniques use only horizontal and vertical wires to complete the connection of all routing nets in the channel. Restricting the wires to only perpendicular directions is often called the Manhattan channel routing model. As VLSI technology advances, the fabrication process does not preclude a layout style in a non-Manhattan routing model. Figure 1 presents the difference between both models. The non-Manhattan channel routing model always uses fewer routing tracks than a Manhattan router. Practically, vertical and horizontal constraint graph no longer exist. Furthermore, the non-Manhattan model produces a solution having a smaller number of net crossings which leads to a smaller number of vias. The basic concept is to interchange a pair of neighboring nets if they are in the wrong order using two wires, one in the +45° direction and the other in the -45° direction. The remaining nets which are not to be interchanged propagate to the next track over vertical wires.

Wang [4] proposed the mini swap algorithm (shown in Figure 2) as a sorting method to solve the non-Manhattan channel routing problem. A heuristic algorithm for the non-Manhattan channel routing based on the bubble-sort technique has been proposed by Chaudhary and Robinson [1]. Two types of inversion tables were used in Chaudhary's algorithm: left and right inversion tables as shown in the example of Figure 3. The criteria of sorting direction depend on the numbers of nonzero elements in the left and right inversion table. The sorting direction is from left to right (right-step) if the number of nonzero entries in the left inversion table is greater than in the right inversion table and vice versa. In the case, if the numbers of nonzero entries in the left and in the right inversion tables are equal, it cannot guarantee an optimal solution in term of routing track required. In addition, the time complexity of the algorithm is O ($k\ n^2$), where (k) is the number of tracks and (n) is the number of terminals in a channel.

The optimal bubble sort algorithm in terms of routing tracks for the non-Manhattan channel routing has been proposed by Chen [2]. The optimal sorting sequence is generated by applying left-step sorting and right–step sorting to the nonequivalent sorting vectors in each pass of the bubble-sort. This kind of enumerative algorithm can achieve an optimal solution with minimum number of sorting passes required, but the time complexity is $O(k^2n)$. In the most recent work [3], Chen solves the problem of equal left and right inversion tables present in [1] while improving the time complexity to become O(kn).

Hierarchical bubble-sorting-based non-Manhattan channel routing has been proposed by Yan [5], using divide-and-conquer technology and applying left-swap sorting and right-swap to the sub vectors. The time complexity is O (kn). All the mentioned algorithms solve the channel routing (CR) problem in a two-layer bubble-sorting-based non-Manhattan channel routing. They then integrate a pair of adjacent intermediate permutations into one routing track in a channel for the three-layer routing.

In this paper, we present a three layer bi-directional bubble sort non-Manhattan channel routing algorithm. The time complexity is O (kn). Although this upper bound is similar to previous work [3, 5], our algorithm has a significant improvement in the time complexity as presented in the remainder of the paper. Section 2 provides a mathematical formulation of the channel routing problem while section 3 presents our solution in three layers. In section 4 we compare the various methods with our algorithm. Section 5 presents our solution in five layers. Finally section 6 concludes the paper.

2. Problem definition

A channel is a pair of vectors of non negative TOP and Bottom numbers of the same dimension.
TOP = t (1), t (2), t (3)… t (m).
Bottom = b (1), b (2), b (3)… b (m).

A number between 1 and m is assigned to each terminal. Terminals having the same label i ($1 \le i \le m$) must be connected. In a bubble sort non-Manhattan channel router (BSNMCR) problem we consider a vector $V = (a_1, a_2, \ldots\ldots, a_n)$ and say that a_i , a_{i+1} are not in the proper order if $a_i > a_{i+1}$ for $1 \le i < n$. V is sorted when all its elements

are in the proper order. Using the left and right inversion tables [1] and choosing the step type optimally, we get a minimum swap pass.

We define (R_1,R_2,..R_j) as the right-inversion table, where Rj is the number of elements to the right of j which are smaller than j. And we define a (L_1,L_2,...,L_j) as a left-inversion table here Lj is the number of elements to the left of j which are bigger than j. For example, the permutation (5,4,1,3,6,2)has the left-inversion table (2,4,2,1,0,0)and the right-inversion table (0,0,1,3,4,1). We conclude that the right-step will decrease each nonzero entry in the left-inversion table by 1, and the left-step will decrease each nonzero entry in the right-inversion table by 1.We choose the step to be a right-step if the number of nonzero entries in the left-inversion table is greater than that in the right inversion table and vice-versa.

An optimal Bubble sort structure (OBSS) [2] generated by applying left-step sorting and right–step sorting to the nonequivalent sorting vectors in each pass of the bubble-sort., for example.

V_0 = (7, 3, 2, 8, 6, 5, 4, 1),
V_1 = (3, 2, 7, 6, 5, 4, 1, 8),
V_2 = (1, 3, 2, 7, 6, 5, 4, 8),
V_3 = (1, 2, 3, 4, 7, 6, 5, 8),
V_4 = (1, 2, 3, 4, 5, 7, 6, 8), and
V_5 = (1, 2, 3, 4, 5, 6, 7, 8). An (OBSS) takes 5 swap passes.

An optimal Hierarchical bubble sort non-Manhattan channel router OHBSS [5], is a tree representing an optimal hierarchical bubble-sorting solution for two-layer. It then integrates a pair of adjacent intermediate permutations into one routing track in a channel for three-layer routing.

The example (14,3,4,5,2,8,6,7,1,13,12,9,10,11) is shown in Figure.4.a for a two-layer solution. Figure 4.b integrates the solution to three layers. Two types of inversion tables were used in Chen's algorithm [3]: left major and right minor tables. The criteria of sorting direction depend on the numbers of nonzero elements in the left major and right minor table. The sorting direction is from left to right (right-step) if the number of nonzero entries in the left major table is greater than in the right minor table and vice versa. If left major and right minor table are equal, Chen [3] checks both left major and right minor, in order going from left to right. For instance, given two tables B (1, 0, 1, 1, 0, 1, 1, 1, 0) and C (1, 0, 1, 0, 0, 1, 1, 1, 1) there is a '1' in B and a '0' in C at the fourth location. Hence, B is considered to be greater than C. Chen [3] uses this simple scheme and calls it lexicographical order.

After the decision (left-right) or (right-left), the algorithm integrates a pair of adjacent intermediate permutations into one routing track in a channel to solve the three-layer routing problem. For an example see figure5.

Using a bi-directional bubble sort non-Manhattan channel routing, there is no need to solve the two layer problem first in order to reach a solution to the three-layer problem. Our algorithm solves a channel routing problem based on three layers

directly. The algorithm is simpler than the ones presented previously. It has no inversion tables, no left or right swap tables, and no major and minor tables.

3. The Bi-directional bubble based non-Manhattan channel routing.

Bi-Directional Bubble Sort works much like Bubble Sort except that it moves from top to bottom and then from bottom to top, swapping items to keep the larger ones towards the bottom and the smaller ones towards the top.

3.1. The algorithm for three layers Bi-Directional Bubble Sort.

```
Algorithm (1)
{
 Top = (1, 2, 3… n) /* top vector */
Bot = input-vector          /* bottom vector */
  while (Bot ≠ Top)
     {
      for (i =1; i < n; i++)
         {
             if (Bot [i] > Bot[i+1])
           Swap (Bot [i], Bot [i+1])
          }
        for (i =n-1; i > 1; i--)
         {
          if (Bot [i+1] < Bot[i])
           Swap (Bot [i], Bot [i+1])
          }
    Write (Bot)
     }
   }
```

It can be found that, the time complexity of using the proposed bi-directional bubble-sort algorithm to route a three-layer non-Manhattan channel is $T_{all}(n) = O(kn)$time, where k is the number of sorting passes required and n is the number of two-terminal nets in a channel.

For example, the vector

(14,3,4,5,2,8,6,7,1,13,12,9,10,11),

is sorted in only three steps as:

(1,3,4,5,2,8,6,7,9,13,12,10,11,14),
(1,2,3,4,5,6,7,8,9,10,12,11,13,14),
(1,2,3,4,5,6,7,8,9,10,11,12,13,14).

Figure 6 shows the resulting tracks.

3.2. The algorithm for five layers Bi-Directional Bubble Sort.

```
Algorithm (2)
{
 Top = (1, 2, 3… n) /* top vector */
Bot = input-vector          /* bottom vector */
     Step =0
while (input-vector ≠ Top)
      {
       for (i =1; i < n; i++)
          {
            if (Bot [i] > Bot[i+1])
             Swap (Bot [i] , Bot[i+1])
            }
         for (i =n; i > 1; i--)
          {
            if (Bot [i+1] < Bot[i])
             Swap (Bot [i] , Bot[i+1])
            }step++
             if (step even)
Write (Bot)   /*not use all the bottom matrix, but only even order of sorting sequence*/
  }
}
```

As in the case of three layer the time complexity T_5 (n) of using our bi-directional bubble-sort algorithm to route a five-layer non-Manhattan channel is O (kn) time.

Taking the previous example of:

(14,3,4,5,2,8,6,7,1,13,12,9,10,11),
(1,2,3,4,5,6,7,8,9,10,12,11,13,14),
(1,2,3,4,5,6,7,8,9,10,11,12,13,14).

It is sorted in only two steps. Figure 7 shows the resulting tracks.

4. Experimental results

Our bubble-sort channel router, Chaudhary's [1] router, and Chen's [3] router have been implemented in C++ language and tested on Intel P4 processor 2.4 GHZ, 256 M RAM. The experimental results are compared and analyzed as follows. First, the comparisons of complexity among three-layer the algorithms are listed in Table (1), next the results of running different routers in worst case are shown in Table (2). Chaudhary's [1] router has a time complexity of $O(kn^2)$ on the average, and $O(n^3)$ in the worst case (i.e. the case where all elements in a sorting vector are in reverse order). In general this router cannot generate an optimal solution in terms of routing tracks required and time performance. Chen [3] has presented an optimal router in terms of routing tracks required for the non-Manhattan channel routing.

Both Chen [3] and our proposed algorithm spend O(kn) on the average and $O(n^2)$ in the worst case. However, our router gives a better absolute time as shown in Table (2).

5. Conclusion

To implement three layers routing, previous research used two layer techniques and integrating a pair of adjacent intermediate permutations into one routing track in a channel for three layers. Our algorithm solves the three layer problem directly. The algorithm is simpler than the ones presented previously. It has no inversion tables, no left or right swap tables, and no major and minor tables.
Table 3 gives a comparison between all routers algorithm.

7. References

CHAUDHARY, K., and ROBINSON, P. Channel routing by sorting. IEEE Transactions on Computer-Aided Design Integrated Circuits Systems. 10:754-760, 1999

CHEN, C.Y.R., HOU, C.Y., and SINGH, U. Optimal algorithms for bubble sort based non-Manhattan channel routing. IEEE Trans. Computer Aided Des. Integrated Circuits Syst. 13:603-609, 1994

CHEN,S.–S.,C–H. Yang and S.–J. Chen. Bubble-sort approach to channel Routing. IEE Proc.-Comput.Digit.Tech. 6:415-422, 2000

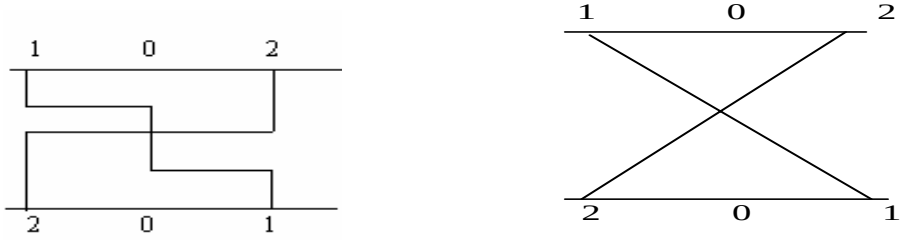
WANG, D.C. Novel Routing Schemes For IC layout part I: Two-layer channel routing. 28th ACM/IEEE Design Automation Conference.49-53,1991

YAN**,** J.–T. Hierarchical bubble-sort-based non-Manhattan channel routing . IEE Proc.-Comput.Digit.Tech. 4:215-220 2000

YAN, J.-T. An improved optimal algorithm for bubble-sorting-based non-Manhattan channel routing. IEEE Trans. Comput. - Aided Des Integi: Circuits Sys. 2:163-171, 1999

YAN, J.-T. Optimal via minimization by selection of layer assignment and routing ordering in a Bubble-sorting-based non-Manhattan channel. IEE Proc.-Computer. Digit. Tech. 1:21-27, 2003

Yongbin Yu, Bo Yang, Xuliang Ahang, and Juebang Yu. Crosstalk Minimization in Four-Layer Non-Manhattan Channel Routing. ICCCAS 2004. 2004 International Conference on, Communications, Circuits and Systems. 2:1281-1285, 2004



(a) (b)

Figure 1. a) Manhattan Dogleg routing. b) Non-Manhattan routing.

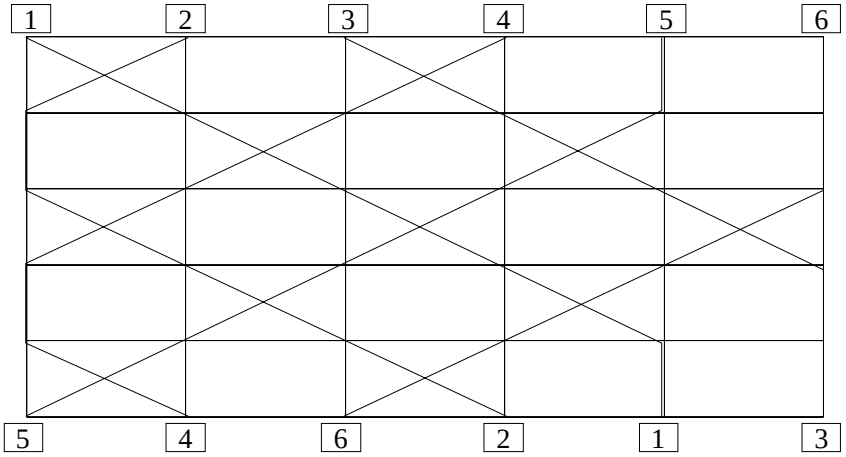


Figure 2. Mini swap router.

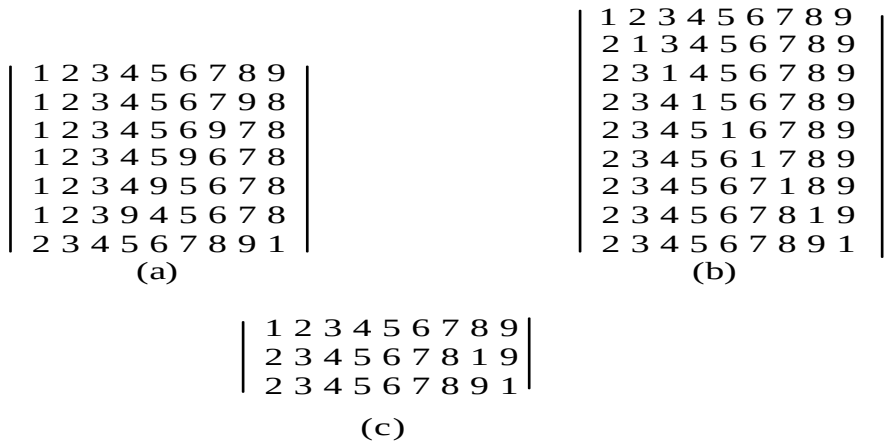


Figure 3. Left or right scanning bubble sort Chaudhary and Robinson a) Left swap only. b) Right swap only. C) Left and right swap passes

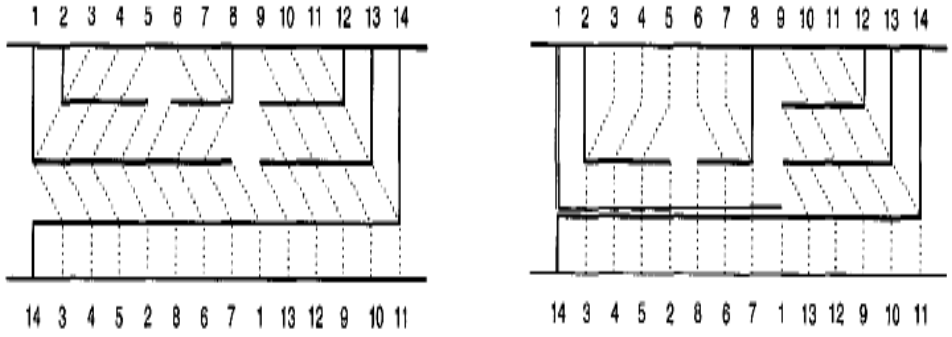


Figure 4. An optimal routing result for [7] a) Two layer Routing result with 'divide and conquer' HBSNMCR b) Three layer Routing result with 'divide and conquer' HBSNMCR

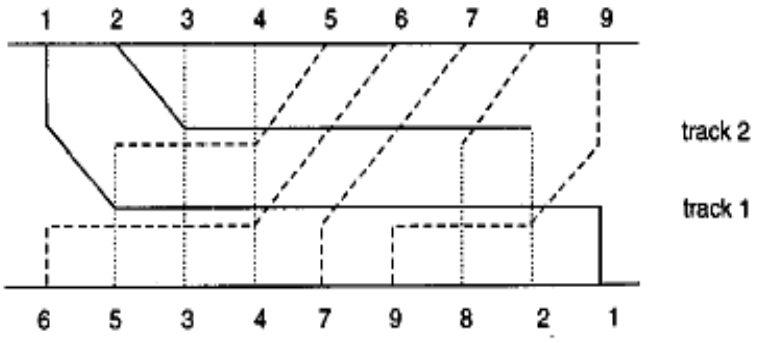


Figure 5. Three-layer non-Manhattan channel problem solved by Chen. [6].

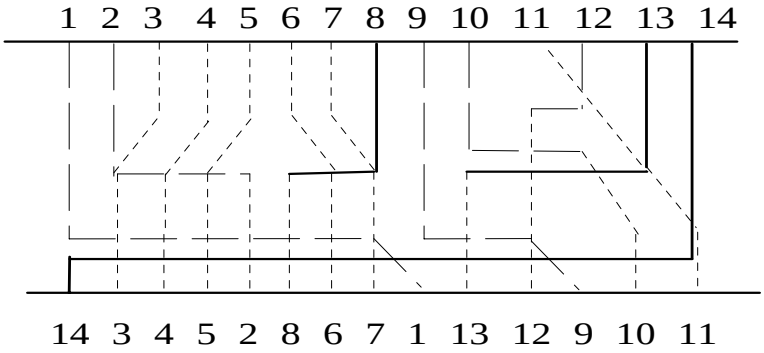


Figure 6. Bi-directional three layer non-Manhattan channel router.

Algorithm	**Time complexity**	
	Average	**worst**
Chaudhary	**O(kn^2)**	**O(n ^{3})**
Chen	**O(kn)**	**O(n ^{2})**
Our algorithm	**O(kn)**	**O(n ^{2})**

Table 1 Comparison for three layer routing algorithm

No of nets	Time(ms)		
	Chaudhary [4]	Chen [6]	Our algorithm
500	1.53	1.06	0.240
	1.56	1.03	0.290
	1.47	1.02	0.210
Average	1.52	1.036	0.246
600	2.33	1.44	0.36
	2.31	1.55	0.400
	2.44	1.44	0.340
Average	2.36	1.47	0.36
700	3.32	1.87	0.45
	3.50	1.89	0.44
	3.45	1.90	0.38
Average	3.42	1.88	0.42
800	4.77	2.39	0.59
	4.64	2.37	0.56
	4.98	2.44	0.58
Average	4.79	2.40	0.57
1000	8.19	3.69	0.92
	8.38	3.63	0.90
	8.13	3.50	1.060
Average	8.23	3.60	0.96

Table 2 Comparison with other routers, each experimental is performed three times and the average is reported.

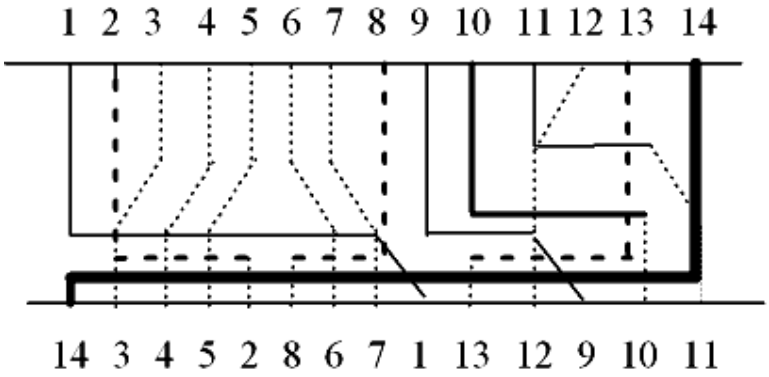


Figure 7. Bi-directional five layer non-Manhattan channel router.

Reference number	Time complexity		Number of steps to reach the final solution
	normal	worst	
[4]	O(k n^2)	O(n^3)	1) left inversion table 2) right inversion table 3) comparison between step number 1,2 4) sorting as a two layer 5) apply LRV=RLV rule 6) sorting as a three layer
[5]	O(k^2 n)	O(n^3)	1) left inversion table 2) right inversion table 3)select the optimum order 4) sorting as a two layer 5) apply LRV=RLV rule 6) sorting as a three layer
[7]	3*k* (n)= O(k n)	O(n^2)	1) left inversion table 2) right inversion table 3) comparison between step number 1,2 4) Divide 5) sorting as a two layer(conquer) 6) apply LRV=RLV rule 7) sorting as a three layer
[6]	3*k*(n)	O(n^2)	1) left major table 2) right minor table 3) comparison between step number 1,2 4) lexicographical table 5) sorting as a two layer 6) apply LRV=RLV rule 7) sorting as a three layer
Our algorithm	O(k n)	O(n^2)	1) left bubble sort 2) right bubble sort

Table 3 Comparison between all routers algorithm