Generating More Compactable Channel Routing Solutions[1]

Jingsheng Cong
Department of Computer Science
University of Illinois, Urbana, IL 61801

D. F. Wong
Department of Computer Science
University of Texas, Austin, TX 78712

Keywords: channel routing, compaction, VLSI layout design, algorithms.

Abstract

In this paper, we present several powerful techniques which allow us to generate efficiently channel routing solutions which are beneficial to further compaction. Our techniques can be applied to straight track compaction as well as to contour routing compaction, and produce very encouraging results. In particular, for Deutsch's Difficult Example, we obtained a straight track routing solution whose area is 7% less than the best known result after straight track compaction. Our router also generated a routing solution which leads to a very satisfactory result after contour routing compaction.

1. Introduction

In VLSI layout design, a significant portion of chip area is used for channel routing. The two-layer grid-based channel routing problem has been well formulated and studied extensively. There are several grid-based channel routers which can consistently produce channel routing solutions which are at most one or two tracks within optimal solutions [De76, YoKu82, RiFi82, BuPe83, ReSS85]. A further study [De85] showed that the routing solutions of these grid-based router could be compacted to obtain a 15% - 20% area reduction. Both straight track compaction [De85, WoLi86, Ch86] and contour routing compaction [De85, XiKu87, Ro87, ChDe88] have been investigated. Usually these compaction algorithms are used as a post-processing step after the grid-based routing solution has been generated. From the experimental results of different channel compactors, it was observed that the area reduction is closely related to the grid-based routing solutions we applied. For example, as reported in [De85], the same contour channel compaction algorithm was applied to three optimal 19-track grid-based routing solutions of Deutsch's Difficult Example. Three different results were obtained. Since all grid-based channel routers were designed to minimize the number of tracks used and do not take the later compaction step into account, it becomes an important problem for channel routing compaction, as raised in [De85] and [XiKu87], to obtain more compactable channel routing solutions. In this paper, we shall present several efficient techniques which allow us to transform grid-based channel routing solutions systematically into more compactable routing solutions.

---

[1] An extended abstract of this paper was presented in DAC-88 [CoWo88].

Usually, contour routing compaction yields significant reduction in channel routing area. For example, up to 23% area reduction was achieved in [ChDe88] for Deutsch's Difficult Example. However, there are several potential problems with contour routing compaction. In routing solutions after contour routing compaction, we have extra wire bends and long wires with only the minimum separation. These increase the routing capacitance and decrease the yield of chip production. On the other hand, straight track compaction results in less area reduction. However, the final compacted routing solutions are much simpler. In many cases, the designer has to make a choice to do either straight track compaction or contour track compaction depending on the specific requirements of the circuit being designed. Thus, a practical channel routing compactor should be able to perform either straight track compaction or contour routing compaction. In this paper, we show how to generate more compactable routing solutions either for straight track compaction or for contour routing compaction. In general, the methods presented in this paper can be used as a preprocessing step for channel compaction.

2. Formulation of the Problem

In grid-based channel routing, we assume that there is a grid structure superimposed on the routing area. Moreover, we assume that wires and vias are dimensionless. A typical grid-based routing solution is shown in Fig. 2-1. When we come to channel compaction, we are no longer restricted to the grid structure, and we must also take the dimension of wires and vias into account. Usually, the width of a via is larger than the width of a wire. Wires or vias from two different nets need to keep a minimum distance, called *feature separation*. The value of the feature separation may be different in different design rules. Basically, there are two types of channel compaction methods. Given a grid-based routing solution S, if we require that all the wires on the same track in S remains in the same straight line after compaction, as shown in Fig. 2-2, we call the compaction *straight track compaction*. (However, the track spacings are not the same constant after straight track compaction.) If we allow wires in S to bend arbitrarily (but still complying with design rules) in the compacted solution, as shown in Fig. 2-3, we call the compaction *contour track compaction*. Let S and S' be two grid-based channel routing solutions for the same channel routing problem. If S' uses less routing area than S after the same channel

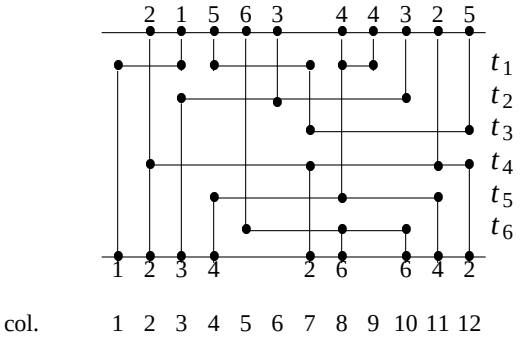


Fig. 2-1 An Example of Two Layer Grid-based Channel Routing Solution

compaction procedure , we say that S' is *more compactable* than S for the channel compaction procedure . In this paper, we show how to obtain more compactable routing solutions for straight track compaction and more compactable routing solutions for contour track compaction.

We use the same set of design rules as presented in [De85]:

path width	1.0
feature separation	1.0
size of via	2.0×2.0
terminal spacing	4.0

This makes comparisons with other compaction results easier by using the same set of design rules. The methods presented in this paper are design rule independent.

In this paper, we assume that there are two layers available for channel routing. One layer is reserved for horizontal wires and the other layer is reserved for vertical wires. A *via* is used to connect wires on different layers. Two vias are *adjacent* if they are in the same column on two adjacent tracks and the two vias belong to two different nets. We define the *height* of a channel to be the distance from the bottom edge of the channel to the top edge of the channel. Usually, we assume that the grid-based routing solutions do not use extra columns at the end of the channel. Therefore, minimizing the channel area after channel compaction is equivalent to minimizing the channel height after channel compaction.

The rest of the paper is organized as follows: In Section 3, we present an algorithm to generate more compactable routing solutions for straight track compaction. In Section 4, we

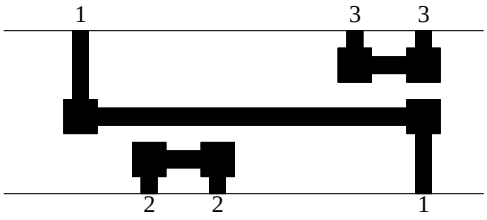


Fig. 2-2 Straight Track Compaction

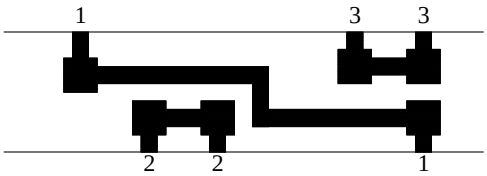


Fig. 2-3 Contour Routing Compaction

show an algorithm to generate more compactable routing solution for contour routing compaction. Experimental results obtained by both algorithms are shown in Section 5.

3. Straight Track Compaction

Straight track compaction is usually achieved by allowing variable track spacing and via offset. If we use uniform track spacing, the track spacing should be 3.0, since there may be two adjacent vias between two tracks (Fig. 3-1 (a)). However, if we allow variable track spacing, we can reduce the track spacing between these two tracks to 2.5, when there are no adjacent vias between the two tracks (Fig. 3-1 (b)). Furthermore, if we allow via offset, we can even reduce the track spacing when there are adjacent vias between two track. For example, if we can shift vias up or down by 0.1 as in [De85], we may reduce the track spacing between two tracks to 2.8 even if adjacent vias exist between these two tracks (Fig. 3-1 (c)). Clearly, existence of adjacent vias in the grid-based routing solution increases the channel height and make routing geometry more complicated after straight track compaction. In a grid-based routing solution, we call two adjacent tracks with adjacent vias between them a *conflicting track pair*. For example, in the grid-based routing solution shown in Fig. 2-1, track 1 and 2 are a conflicting track pair since there is a pair of adjacent vias at column 3 on these two tracks. It is easy to show that for a grid-based routing solution without conflicting track pair, the minimum channel height can be achieved after straight track compaction using uniform track spacing without via offset (a much simpler routing geometry!). Thus, the key problem for generating more compactable routing solution for straight track compaction is to minimize the number of conflicting track pairs.

We introduce two powerful and efficient techniques to transform a grid-based routing solution into another solution with the minimum number of conflicting track pairs. One technique is *track permutation*. For the example shown in Fig. 2-1, there are 4 conflicting track pairs. If we exchange track 2 and 3, we obtain a routing solution with only 2 conflicting track pairs (Fig. 3-2) Another technique we use is *local re-routing*. For the routing solution we obtained in Fig. 3-2, if we reroute net 4 at column 8 and 9, we can remove adjacent vias between track 5 and 6. Thus, we obtain a routing solution with only one conflicting track pair (Fig. 3-3).

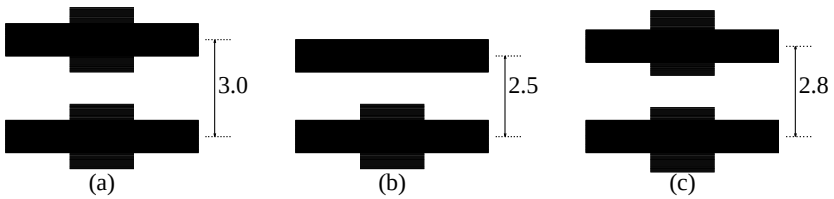


Fig. 3-1 Variable Track Spacing and Via Offset

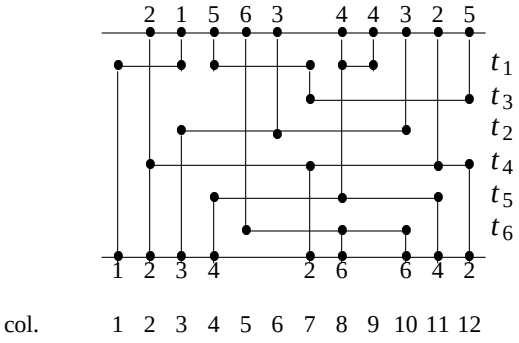


Fig. 3-2 After Exchanging Track 2 and 3 in Fig. 2-1.

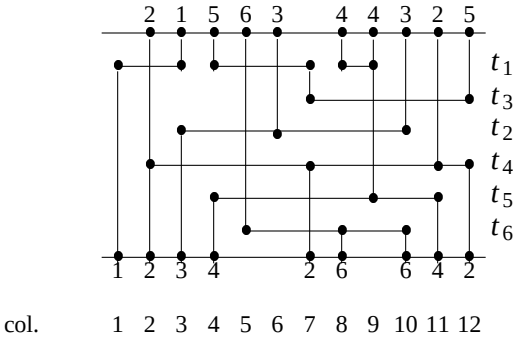


Fig. 3-3 The Solution in Fig. 3-2 after Rerouting.

3.1. Track Permutation

First, we shall discuss how the tracks in a valid two layer routiong solution can be permuted to yield another valid two layer routing solution. Let S be a two layer solution using tracks $t_1, t_2, ..., t_w$. Let π be a permutation on $\{1, 2, \cdots, w\}$. We use $\pi(S)$ to denote a two layer wiring configuration where the $\pi(i)$-th track of $\pi(S)$ is track t_i in S, and at each column there is a vertical wire segment connecting the $\pi(i)$-th track and the $\pi(j)$-th track in $\pi(S)$ if and only if there is a vertical wire segment connecting t_i and t_j in S. (For convenience, we assume that the upper edge of the channel is the 0-th track, and that the lower edge of the channel is the $(w+1)$-th track. Moreover, the upper edge and the lower edge of the channel remain at their original positions under any track permutation π, i.e., $\pi(0) = 0$, and $\pi(w+1) = w+1$.) π is *a valid track permutation* if and only if $\pi(S)$ has no horizontal or vertical overlap of wires from different nets. Since there is no horizontal overlap in S, for any permutation π there will be no horizontal overlap in $\pi(S)$. Thus, it is clear that π is a valid track permutation if and only if $\pi(S)$ has no vertical overlap of wires from different nets. We observe that not every track permutation is valid. For example, exchanging track 3 and 4 in Fig. 2-1 is not a valid track permutation since there would be vertical wire overlap at both column 7 and column 12. To characterize valid track permutations, we define the *track ordering graph* $G(S)$ of a two layer grid-based routing solution

S. $G(S)$ is a directed acyclic graph. Each node in $G(S)$ represents a track in S. There is a directed edge from node t_i to node t_j in $G(S)$ if $i < j$ (track t_i is above track t_j) and there is a via on track t_i in the same column as a via on track t_j. Fig. 3-4 shows the track ordering graph of the routing example in Fig. 2-1. Such an edge specifies the restriction that track t_i and t_j cannot be adjacent, otherwise they form a conflicting track pair. Using the track ordering graph, we have the following characterization:

Lemma 1 For a two layer solution S using w tracks $t_1, t_2, ..., t_w$, if l is a topological labeling on $G(S)$, then the permutation $\pi = \begin{pmatrix} 1 & 2 & \cdots & w \\ l(1) & l(2) & \cdots & l(w) \end{pmatrix}$ is a valid track permutation.

Lemma 2 Let S be a two layer routing solution with only restricted doglegs using tracks $t_1, t_2, ..., t_w$. If π is a valid track permutation, let $l(i) = \pi(i)$ for $1 \le i \le w$, then l is a topological labeling on $G(S)$.

For the proofs of these two lemmas, see [Co87]. The crucial problem left is to obtain a valid track permutation π such that the number of conflicting track pairs in $\pi(S)$ is minimized for a given S. For a valid track permutation π, we say the π is *optimal* if for any other valid track permutation π', the number of conflicting track pairs in $\pi(S)$ is no more than the number of conflicting track pairs in $\pi'(S)$. We can show that the problem of finding an optimal track permutation of a given routing solution S is equivalent to the separation problem for directed acyclic graphs, which is formulated as follows [LeVW84]:

Input: A directed acyclic graph G.

Question: Find a topological labeling of G that minimizes the total number of edges between vertices with consecutive labels.

Theorem 1 Given a grid-based routing solution S without unrestricted doglegs, the problem of finding an optimal track permutation of S is equivalent to the separation problem for $G(S)$.

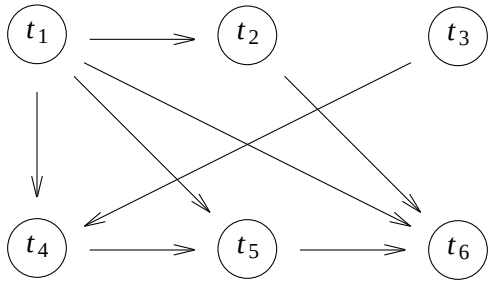


Fig. 3-4 The Track Ordering Graph of the Example in Fig. 2-1.

Proof: Assume that S has w tracks. Since S has no unrestricted doglegs, according to Lemma 1 and Lemma 2, a track permutation $\pi = \begin{pmatrix} 1 & 2 & \cdots & w \\ l(1) & l(2) & \cdots & l(w) \end{pmatrix}$ is a valid track permutation if and only if l is a valid topological labeling of $G(S)$. Moreover, the $l(i)$-th track and the $l(j)$-th track in $\pi(S)$ form a conflicting track pair if and only if $l(i)$ and $l(j)$ are two consecutive numbers and there is an edge between t_i and t_j. Therefore, the number of conflictiong track pairs in $\pi(S)$ is equal to the number of edges between nodes with consecutive labels under the topological labeling l of $G(S)$. Thus, the problem of finding a track permuatation with the minimum number of conflictiong track pairs is equivalent to the problem of finding the topological labeling of $G(S)$ with the minimum number of edges between nodes with consecutive labels. □

Note that when S has unrestricted doglegs, the topological labeling obtained by solving the separation problem of $G(S)$ still gives a valid track permutation of S. However, it may not be optimal since S may have valid track permutations which cannot be induced from a topological labeling of $G(S)$. It has been shown that the separation problem can be solved in polynomial time ([LeVW84], [WoLi86]). We now briefly describe a linear time algorithm for the separation problem. The details of the algorithm can be found in [WoLi86].

Given a directed acyclic graph G, let α be a sequence of vertices in G. We use $G - <\alpha>$ to denote the subgraph of G resulting from removing from G the vertices in α and the edges incident to the vertices in α. We say that a vertex v is *free* in G if v has no incoming edges in G. We define the *level* of a vertex x in G to be the length (number of vertices) of a longest path starting at x, and denote it *level*(x). Let $*$ be a special symbol different from all the vertices in G. A *level labeling* of G is a linear ordering of the vertices of G (together possibly with some $*$ $'s$) such that it is of the form $\alpha_L x_L \alpha_{L-1} x_{L-1} \cdots \alpha_2 x_2 \alpha_1$ where :

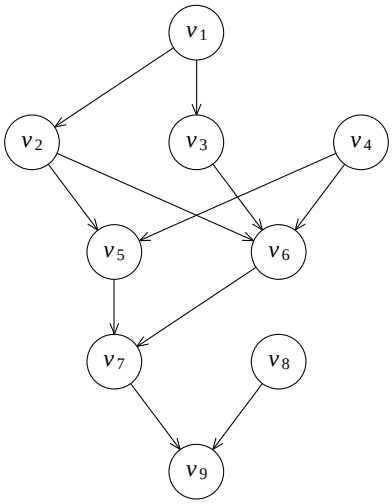
(1) Index L is the length of the longest path in G,

(2) Sequence α_k is a sequence of all the vertices in $G - <\alpha_L x_L \cdots \alpha_{k-1} x_{k-1}>$ of level k in G,

(3) Vertex x_k is either a vertex in G of level less than k or is the special symbol $*$,

(4) If x_k is not $*$, then x_k is free in $G - <\alpha_L x_L \cdots \alpha_k>$ and is not adjacent to the last vertex in α_k in G.

Intuitively, in a level labeling, we order the vertices in G level by level. We do not allow the last vertex of level k to be adjacent with the first vertex of level $k-1$. When we can not avoid this, we insert a free vertex of level less than $k-1$ or the special symbol $*$ between these two vertices. A level labeling can be characterized by its *jump sequence* (level(x_L), level(x_{L-1}), ..., level(x_2)) where level($*$) is defined to be 0. Jump sequences are compared using the lexicographic order. Thus, for two jump sequences $(t_1, \ldots, t_{L-1})$ and $(s_1, \ldots, s_{L-1})$ we say that $(t_1, \ldots, t_{L-1}) > (s_1, \ldots, s_{L-1})$ if for some j, $1 \le j \le L-1$, $t_i = s_i$ for $1 \le i < j$ and $t_j > s_j$. A *HLF labeling* (Highest Level First) of G is a level labeling whose jump sequence is lexicographically as large as possible. Fig. 3-5 shows an example to illustrate the above definitions. In Fig. 3-5, the vertices



G

L = 5

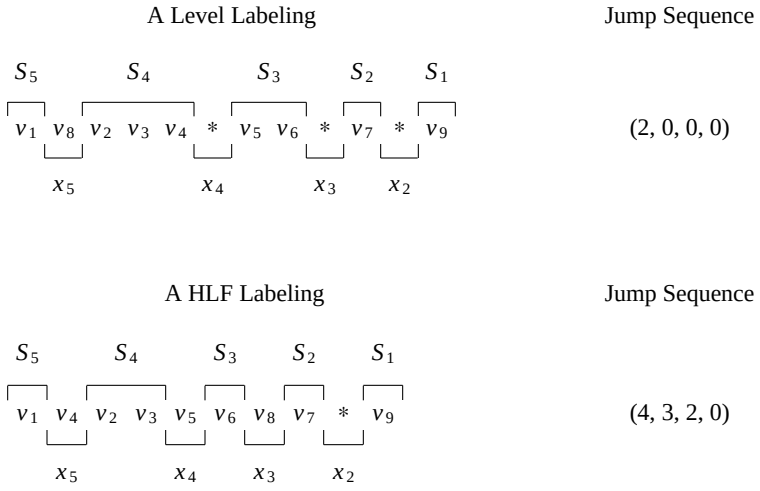


Fig. 3-5 Level labeling, HLF labeling, and jump sequence.

in the given graph are arranged according to their levels. We show two level labelings and their jump sequences. It is not difficult to verify that the second level labeling is a HLF labeling since its jump sequence is the lexicographically largest among all possible level labelings. There are two relevant results in [WoLi86]. The first result specifies the relation between HLF labelings

and the optimal solutions to the separation problem.

Theorem 2 Given a directed acyclic graph G, the topological labeling resulting from removing all the $*$'s from a HLF labeling of G is an optimal solution to the separation problem of G.

The second result shows that HLF labelings can be computed efficiently.

Theorem Given a direct acyclic graph G, we can find a HLF labeling of G in $O(n+m)$ time, where n in the number of vertices in G and m is the number of edges in G.

The proofs of these two theorems can be found in [WoLi86]. It is easy to see from these two theorems that the separation problem can be solved in linear time. Therefore, we can obtain the optimal track permutation in linear time.

A similar track permutation technique was used to transform two layer channel routing solutions into three layer channel routing solutions in [CoWL87], in which the problem of finding the optimal track permutation was reduced to the Two Processor Scheduling Problem.

3.2. Local Re-routing

Local re-routing is performed after we permute the tracks of the channel routing solution according to the optimal track permutation. By re-routing some nets locally, we can remove some adjacent vias, and thus further reduce the number of conflicting track pairs. This task is accomplished by using a maze router, which works as follows: For the partial routing solution in Fig. 3-6 (a), we remove via v_1 (Fig. 3-6 (b)) and try to connect the portion of the net containing x with the portion of the net containing the horizontal segment h_1 (Fig. 3-6 (c)). We try to remove via v_2 in a similar way if the removal of via v_1 fails. Our maze router is based on the classical wave propagation algorithm of Lee [Le61].

In our maze router, we sometimes allow a short vertical wire to be put on the horizontal layer, or a short horizontal wire to be put on the vertical layer. Also, when the original routing solution is very crowded, local re-routing may not be very effective to remove adjacent vias. In

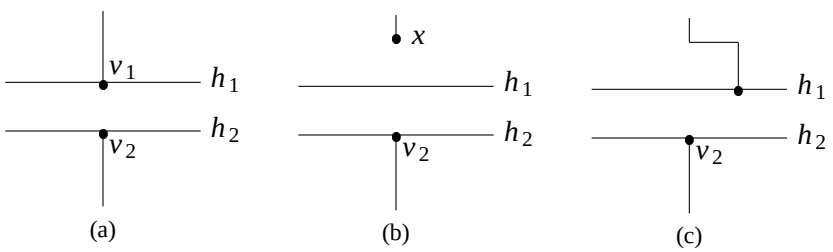


Fig. 3-6 Local Rerouting

this case, we try to insert an empty track at the most crowded area. Although increasing the number of tracks tends to increase the channel height, we might be able to reduce the number of conflicting track pairs significantly by using the empty track for local re-routing again. Thus, the compacted channel height may decrease. Our maze router will generate both routing solutions (one with empty tracks and the other one without empty tracks) and choose the one with smaller channel height.

Combining the track permutation step and the local re-routing step, Fig. 3-7 shows our algorithm to generate more compactable channel routing solution for straight track compaction.

4. Contour Routing Compaction

Contour routing compaction is usually done by processing tracks one by one from the bottom of the channel to the top of the channel. For each track, we assign wires and vias on the track to the lowest possible position. Since we do not have to keep distance between two tracks equal, adjacent vias are no longer the critical consideration. However, the phenomenon shown in Fig. 4-1, referred to as "bump propagation" in [De85] may affect the final compacted result. Without loss of generality, assume that we compact the channel from bottom to top. Then, a via on a track near the bottom of the channel may cause adjacent tracks to jog around it and propagate all the way up to the top. Thus, we want to find routing solutions such that the number of vias on the tracks near the bottom of the channel is minimized. (Symmetrically, if we compact the channel from top to bottom, then, we want to find routing solutions such that the number of vias on the tracks near the top of the channel is minimized.) For example, in Fig. 4-2 (a), the bump propagations of via a and b affect via c. However, after we move via a and b up in Fig. 4-2 (b), their bump propagations do not affect via c anymore. This results in a smaller

Algorithm Solutions_for_straight_track_compaction
1. Input a two-layer grid-based routing solution S;
2. Construct the track ordering graph $G(S)$;
3. Compute the optimal track permutation by solving the Separation Problem for $G(S)$;
4. **Repeat**
Remove adjacent vias as many as possible by local re-routing;
Decide if an empty track needs to be inserted;
Until no reduction of the channel height.

Fig. 3-7 Overall algorithm to generate more compactable routing solutions for straight track compaction.

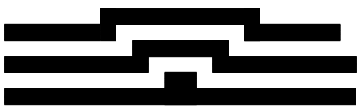


Fig. 4-1 Bump Propagation

channel height. Again, track permutation and local re-routing turn out to be powerful tools to transform standard grid-based routing solutions into solutions of this kind.

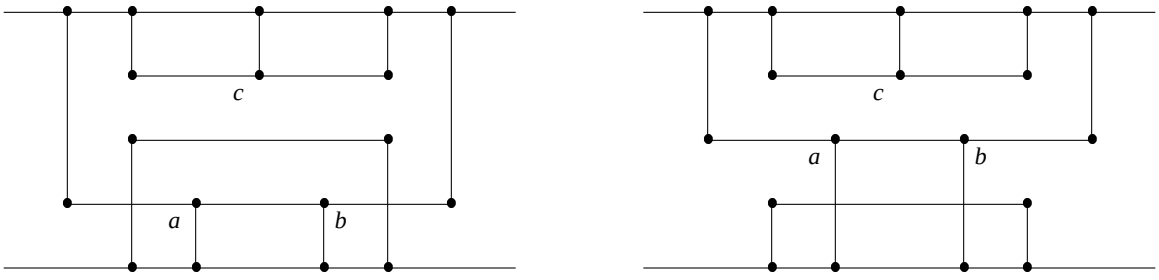
4.1. Track Permutation

We can still use Lemma 1 and Lemma 2 to characterize valid track permutations. However, the definition of an optimal track permutation is different. Assume v_i is the number of vias on track t_i. Let t_i be the i-th track from the top. Then the propagation length caused by all the vias on t_i can be calculated as $i \cdot v_i$, because i measure the distance from track t_i to the top of the channel. Given a valid track permutation π, the bump propagation caused by vias on t_i after applying track permutation π can be calculated as $\pi(i) \cdot v_i$. Thus, we want to find a valid track permutation π such that $b(\pi) = \sum_{i=1}^{n} \pi(i) \cdot v_i$ is minimized, where $\pi(i)$ is the position of track t_i from the top of the channel after permutation π is performed. We say that a valid track permutation π is *optimal* if $b(\pi) \le b(\pi')$ for any other valid track permutation π'. We reduce the problem of finding an optimal track permutation to the following single machine unit job sequencing problem:

Input: n jobs of unit execution time to be processed by a single machine. The precedence constraints is specified by a given directed acyclic graph. Each job t_i has a weight w_i $(i = 1, 2, ..., n)$.

Question: Find a feasible sequence s such that the weighted completion time $c(s) = \sum_{i=1}^{n} w_i \cdot s(i)$ is minimized, where $s(i)$ is the rank of the job t_i in the sequence s (assume that the first job starts at $t = 0$).

Theorem 4 Given a two-layer grid-based channel routing solution S using w tracks with only restricted doglegs, assume that v_i is the number of vias on track t_i, we construct an instance I of the single machine unit job sequencing problem as follows: I consists of w jobs with the



(a) (b)

Fig. 4-2 Minimizing bump propagation.

track permutation graph $G(S)$ as the precedence constraint graph. The weight of job t_i is v_i. Then the problem of finding an optimal track permutation of S is equivalent to the single machine unit job sequencing problem of I.

Proof: According to Lemma 1 and Lemma 2, a track permutation π is valid if and only if π is a topological labeling of $G(S)$. Moreover, s is a feasible sequencing of I if and only if s is a topological labeling of $G(S)$. Therefore, there is a ono-to-one correspondence between a valid track permutation π of S and a feasible sequencing s of I such that $\pi(i) = s(i)$ for $1 \le i \le w$. Clearly, $b(\pi) = c(s)$ since the weight of job t_i equals the number of vias v_i on track t_i. Thus, the problem of finding a valid track permutation of S with the minimum bump propagation is equivalent to the problem of finding a job sequencing of I with the minimum weighted completion time. □

This sequencing problem has been proved to be NP-Complete [La78]. However, we can show that in our case for most of the problems, the number of tasks is small (since it is equal to the number of tracks in the routing solution, usually it is no more than 50). Moreover, the number of feasible sequencings is quite limited in most of the cases since $G(S)$ is quite dense. Thus, branch and bound algorithm can obtain the optimal result quickly. Also, Sidney's decomposition theorem [Si75] can be used to find the optimal solution.

4.2. Local Re-routing

In this step, instead of minimizing the number of adjacent vias as in Sec. 3.2, we try to re-route some of the nets to remove unnecessary vias by a maze router. This will help to minimize the final compacted channel height, since each via will cause 'bump propagation'. Since we do not want to introduce new vias when we do re-routing, our maze router does not switch layers. Also, since removal of some of the vias may block the removal of other vias, and we want to minimize the number of vias at the bottom of a channel as much as possible because of 'bump propagation', our maze router will try to remove vias from the bottom of the channel to the top of the channel. We use a similar wave propagation algorithm as presented in the preceding section to implement our maze router.

Algorithm Solutions_for_contour_track_compaction.
1. Input a two-layer grid-based routing solution S;
2. Construct the track ordering graph $G(S)$;
3. Compute the optimal track permutation by solving the corresponding single machine unit job sequencing problem.
4. **Repeat**
   Remove vias from bottom to top by local re-routing;
   Decide if an empty track needs to be inserted;
   Until no reduction of the channel height.

Fig. 4-3 Overall algorithm to generate more compactable routing solutions for contour track compaction.

Combining the track permutation step and the local re-routing step, Fig. 4-3 shows our algorithm to generate more compactable channel routing solution for contour track compaction. It is very similar in structure to the algorithm to generate more compactbale channel routing solution for straight track compaction in the last section.

5. Experimental Results

We implemented our algorithms for generating more compactable routing solutions both for straight track compaction and for contour routing compaction respectively. Our programs are written in the Pascal language running under Unix 4.3BSD on a Pyramid machine. Table 5-1 shows the routing solutions we obtained for straight track compaction. YK3a, YK3b and YK3c are examples 3a, 3b and 3c, respectively in Yoshimura and Kuh's paper [YoKu82]. D1, D3 and D5 are from the GTE Layout published in [De76]. Diff is the famous Deutsch's Difficult Example. Our router removed all the adjacent vias without inserting empty tracks for all the examples except Deutsch's Difficult Example. Thus, after straight track compaction, we achieved the minimum channel height using uniform track spacing. For Deutsch's Difficult Example, we obtained two routing solutions. Both have a smaller channel height than the best reported result (of height 54.6 [De85]) based on the same set of design rules after straight track compaction. The first one has 4 conflicting track pairs, and its channel height is 7% smaller than the best reported result after straight track compaction with variable track spacing and via offset. The second one resolved all the conflicting track pairs by inserting an empty track (Fig. 5-1). It has a slightly larger channel height but can be implemented by uniform track spacing after straight compaction. It is the first 20 track solution without adjacent vias for Deutsch's Difficult Example ever reported. Running time of our program for straight track compaction on all the tested examples is less than 50 CPU seconds.

Table 5-2 shows the routing result of Deutsch's Difficult Example we obtained for contour routing compaction and comparisons with other results. Our router generated three different routing solutions based on three different initial solutions to Deutsch's Difficult Example. Our1

Ex.	d	# of tracks	# of conflicting track pairs	channel height after compaction	lower bound for channel height
YK3a	15	15	0	39	39
YK3b	17	17	0	44	44
YK3c	18	18	0	46.5	46.5
D1	18	18	0	46.5	46.5
D3	15	15	0	39	39
D5	17	17	0	44	44
Diff	19	19	1	50.7	49
Diff	19	20	0	51.5	49

Table 5-1 Experimental Results for Straight Track Compaction.

is based on the 19-track solution by the hierarchical channel router in [BuPe83], the channel height of Our1 after compaction is 45.0. Our2 is based on the 20-track solution by Yoshimura and Kuh [YoKu82], the channel height of Our2 after compaction is 44.0. Our3 is based on the 19-track solution by Deutsch [De85], the channel height of Our3 after compaction is 43.0. The emphasis of our work is to generate more compactable grid-based channel solutions. The compaction step is carried out by a straight forward one dimensional channel compactor. It is possible that the results quoted above can be further improved by using a more intelligent channel compactor. In [ChDe88], a more sophisticated channel compactor was presented, in which both via minimization and via shifting in horizontal direction are applied. The channel height of the routing solution in [De85] after compaction is 42.0. Running time of our program for contour channel routing is less than 12 CPU minutes on all the tested examples.

6. Remarks and Conclusions

In this paper, we showed how to generate grid-based channel routing solutions which are beneficial to later compaction. The basic idea is to do solution transformation based on routing solutions produced by good channel routers. Compaction results based on our solution have not only a smaller channel height but also a simpler routing geometry. Running time for our algorithms is quite short, and therefore, our program can be used efficiently as a pre-processing step of channel routing compaction to improve compaction results greatly.

7. Acknowledgements

We thank Prof. C. L. Liu for his valuable suggestions. We thank Ron Libeskind-Hadas for his helpful comments on the original manuscript. The first author is partially supported by National Science Foundation under grant MIP 87-03273, by the Semiconductor Research

Based solution	# of tracks	channel height after compaction
[De76]	21	49.0
[RiFi82]	20	48.0
[YoKu82]	20	47.0
[ReSS85]	19	47.0
[BuPe83]	19	46.0
[Ro87]	81	46.0
[De85]	19	45.0
Our1	19	45.0
Our2	20	44.0
Our3	19	43.0
[ChDe88]	19	42.0

Table 5-2 Comparisons of Contour Routing Compaction on Deutsch's Difficult Example

Cooperation under contract 87-DP-109, and by a grant from General Electric Company. The second author is partially supported by the Texas Advanced Research Program under grant 4096, by the National Science Foundation under grant MIP-8909586, and by an IBM Faculty Development Award.

References

[BuPe83] Burstein, M. and R. Pelavin, ‘‘Hierarchical Channel Router’’. *Integration, the VLSI journal* (1983) Vol. 1, pp. 21-38.

[Ch86] Ng, C. H., ‘‘An Industrial World Channel Router For Non-rectangular Channels’’. *Proc. 23th Desisgn Automation Conf.* (1986) pp. 490-494.

[ChDe88] Cheng, C. K. and D. N. Deutsch, ‘‘Improved Channel Routing by Via Minimization and Shifting’’. *Proc. 25th Desisgn Automation Conf.* (1988) pp. 677-680.

[Co87] Cong, J., ‘‘A New Approach to Three Layer Channel Routing’’, MS-Thesis, Dept. of Computer Sci., Univ. of Illinois, Urbana, Illinois, May, 1987.

[CoWL87] Cong, J., D. F. Wong and C. L. liu, ‘‘A New Approach to the Three Layer Channel Routing Problem’’. *Proc. ICCAD-87* (1987) pp. 378-381.

[CoWo88] Cong, J. and D. F. Wong, ‘‘How to Generate More Compactable Channel Routing Solutions’’. *Proc. ACM/IEEE 25th Design Automation Conf.* (1987), pp. 663-666.

[De76] Deutsch, D. N., ‘‘A Dogleg Channel Router’’. *Proc. 13th Desisgn Automation Conf.* (1976) pp. 425-433.

[De85] Deutsch, D. N., ‘‘Compacted Channel Routing’’. *ICCAD-85* (Nov., 1985) pp. 223-225.

[La78] Lawler, E. L., ‘‘Sequencing Jobs to Minimize Total Weighted Completion Time Subject to Precedence Constraints’’. *Annals of Discrete Mathematics* (1978) Vol. 2, pp. 75-90.

[Le61] Lee, C. Y., ‘‘An Algorithm for Path Connection and its Application’’. *IRE Trans. on Electronic Computers* (1961) Vol. EC-10, pp. 346-365.

[LeVW84] Leung, J., O. Vornberger and J. D. Witthoff, ‘‘On some variants of the bandwidth minimization problem’’. *SIAM J. Comput.* (1984) Vol. 13, No. 3, pp. 650-667.

[ReSS85] Reed, J., A. Sangiovanni-Vincentelli and M. Santomauro, ‘‘A New Symbolic Channel Router: YACR2’’. *IEEE Trans. on Computer Aided Design of CIAS* (1985) Vol. CAD-4, No. 3, pp. 208-219.

[RiFi82] Rivest, R. L. and C. M. Fiduccia, ‘‘A ’Greedy’ Channel Router’’. *Proc. 19th Design Automation Conf.* (1982) pp. 418-424.

[Ro87] Royle, J. et al, ‘‘Geometrical Compaction in One Dimension for Channel Routing’’. *Proc. of 24th Design Automation Conf.* (June, 1987) pp. 140-145.

[Si85] Sidney, J. B., ‘‘Decomposition Algorithms for Single-Machine Sequencing with Precedence Relations and Deferral Costs’’. *Operations Research* (1975) Vol. 23, No. 2, pp. 283-298.

[WoLi86] Wong, D. F. and C. L. Liu, ‘‘Compacted Channel Routing with Via Placement

Restriction''. *Integration, the VLSI journal* (1986) Vol. 4, pp. 267-307.

[XiKu87] Xiong, X. and E. S. Kuh, ''Nutcracker: An Efficient and Intelligent Channel Spacer''. *Proc. of 24th Design Automation Conf.* (1987) pp. 298-304.

[YoKu82] Yoshimura, T. and E. S. Kuh, ''Efficient Algorithms for Channel Routing''. *IEEE Trans. on Computer Aided Design of ICAS* (Jan. 1982) Vol. CAD-1, pp. 25-35.