Purdue University
Purdue e-Pubs

Department of Computer Science Technical Reports | Department of Computer Science

1983

On Bounding the Number of Contact Points in Channel Routing

Susanne E. Hambrusch
Purdue University, seh@cs.purdue.edu

Report Number:
83-438

Hambrusch, Susanne E., "On Bounding the Number of Contact Points in Channel Routing" (1983). *Department of Computer Science Technical Reports.* Paper 359.
https://docs.lib.purdue.edu/cstech/359

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.

On Bounding the Number of Contact Points in Channel Routing

Susanne E. Hambrusch
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD-TR-438 March 1983

Abstract

All known algorithms for the general channel routing problem in the 2-layer knock-knee model with $2d-1$ tracks use $\Theta(dn)$ contact points. We present an algorithm for a restricted class of channel routing problems that uses $2d-1$ tracks and $O(n)$ contact points and runs in time $O(n)$. While channel routing problems of this restricted class were successfully used to prove lower bounds on the channel width, any proof that $2d-1$ tracks and $O(n)$ contact points cannot be achieved simultaneously in general must make use of some special properties not present in the restricted channel routing problem. We supply insight into those properties and into some of the difficulties that must be overcome by an algorithm that uses $O(n)$ contact points for the general channel routing problem.

Key Words

Channel routing, two-terminal nets, density, contact points, knock-knees, wire layout, wiring.

1. Introduction

The *Channel Routing Problem* (CRP) is a wiring problem in integrated circuit design that has received much attention recently ([BB], [BP], [D], [DKSSU], [PL], [RBM]). One common model for channel routing is the 2-layer model with knock-knees: Each layer can be used to run horizontal and vertical wires, and wires on different layers are allowed to cross and to share a corner (i.e., form a knock-knee), but are not allowed to overlap.

The (infinite) channel of width t consists of the grid points (i,j), $0 \le i \le t+1$, $-\infty < j < \infty$, where i is the track number and j is the column number, and the edges connecting adjacent grid points. Grid points on track 0 and $t+1$ are called terminals. A wire is a path connecting adjacent grid points and a wire can switch from one layer to the other by using a grid point as a *contact point*.

In the CRP we are given n (two-terminal) nets (p_i, q_i), where p_i is a column number on track 0 and q_i is a column number on track $t+1$, $1 \le i \le n$, and no two nets share a common terminal. A solution to the CRP consists of the channel width t and the wires connecting the two terminals of each net in the channel. Besides minimizing the channel width, the number of contact points used is an important measure in a channel routing algorithm. In this paper we present an algorithm for a restricted class of CRPs that uses an optimal (within a constant) number of contact points. We also discuss the difficulties that arise when trying to minimize the number of contact points in the general CRP.

A trivial lower bound on the channel width t is the density d of a CRP. The density is the maximum over all x of the number of nets (p_i, q_i) for which $p_i < x < q_i$ or $q_i < x < p_i$. Leighton has improved this lower bound by showing that there exist CRPs of density d that require $2d-1$ tracks [L]. The channel routing algorithms proposed in [BB] and [RBM] produce solutions that use $2d-1$ tracks, but require $O(dn)$ contact points in the worst case.

We first describe a channel routing algorithm for one-sided CRPs that uses $2d-1$ tracks and at most $4n$ contact points. A one-sided CRP consists either of right and trivial nets (right CRP), or of left and trivial nets (left CRP). A net (p_i, q_i) is a *right* net if $p_i<q_i$, a *left* net if $p_i>q_i$, and a *trivial* net if $p_i=q_i$. Lower Bounds on the channel width for this and for another, more restrictive, model have been obtained by considering only right (or left) CRPs ([BR], [L]). Our result thus says that right (or left) CRPs alone are not powerful enough to prove $2d-1$ tracks and $O(n)$ contact points *cannot* be achieved simultaneously. We then show how to extend our algorithm to *weak* CRPs, and give some insight into the difficulties present when trying to achieve $O(n)$ contact points and $2d-1$ tracks for general CRPs.

The 3-layer CRP can be solved more efficiently: Preparata and Lipski present a channel routing algorithm whose solution uses d tracks and $O(n)$ contact points on a 3-layer channel [PL]. We will follow the definitions introduced in [PL] and distinguish between the *wire layout* and the *wiring* of a CRP. Informally, the wire layout describes the path of the wires of the nets without considering the assignment to the layers, while the wiring gives the assignment of wire sections to the layers. See Fig. 1.1, where (b) shows the wiring of the wire layout given in (a).

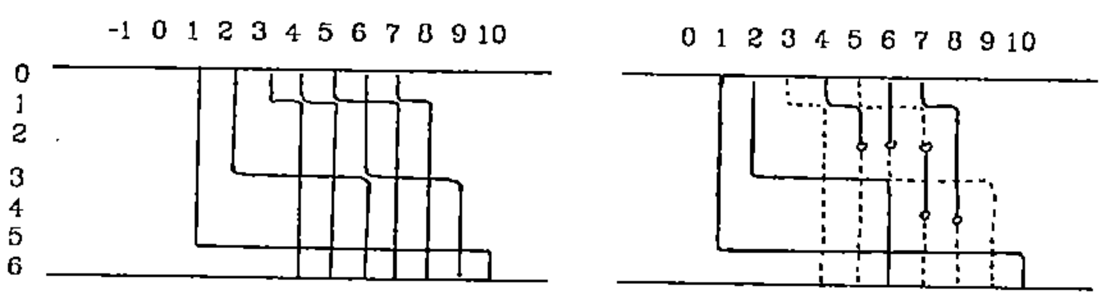


Fig 1.1: (a) wire layout (b) wiring
of the nets (1,10) (2,6) (6,9) (3,4) (4,5) (5,7) (7,8)
which have density 3 and are wired on 5 tracks

The algorithms described in [BB] and [RBM] determine the wire layout of the CRP using d tracks and then interleave the d tracks with $d-1$ tracks. The grid points of the interleaved (even numbered) tracks are used as contact points for vertical wires that need to switch layer in order to cross over a horizontal wire in a track below (see Fig. 1.1 (b)). Thus the shape of a net in the wire layout will be the same as in the wiring. Our algorithm also uses d tracks for horizontal wires and $d-1$ interleaved tracks for layer switches, but it will change the shape of a net in the wiring step according to a set of rules that are shown to minimize the number of contact points.

2. The One-Sided Channel Routing Algorithm

We describe a channel routing algorithm that solves right CPRs in a channel of width $2d-1$ by using at most $4n$ contact points. The algorithm for left CRPs is analogous. We assume that the right CRP does not contain any trivial nets (trivial nets can easily be added after the algorithm), and that the CRP is *full*; i.e., each terminal (excluding the ones on the right and left end of the channel) is either the starting or the ending point of a net. Each right CRP of density d uniquely decomposes into d *right runs*. The k-th right run, $1 \le k \le d$, is a maximal sequence of nets (p_1^k, q_1^k), (p_2^k, q_2^k), ..., $(p_{i_k}^k, q_{i_k}^k)$ such that $p_j^k < q_j^k$ and $q_j^k = p_{j+1}^k$.

The obvious wire layout for right CRP is to assign each run to one of d tracks and to produce a knock-knee when a net ends and a new one starts up; see Fig. 1.1 (a). If the wiring is then obtained by using $d-1$ interleaved tracks to accommodate necessary contact points, up to dn contact points are needed. An even stronger result can be proven: if the wiring of a right CRP consists only of nets having the 'simple' shape (i.e., a vertical, a horizontal, and a vertical wire), $\Omega(dn)$ contact points are required. Thus, when minimizing the number of contact points the wired nets have to have a different shape than the nets in the 'obvious' wire layout.

In our wiring algorithm we refer to the d tracks that run horizontal wires as track 1, ..., track d, while the $d-1$ interleaved tracks are unnamed. We initially assign each run to a numbered track (this uniquely determines the 'obvious' wire layout). Wires running on layer 1 (layer 2) will be called red (blue) wires. When a vertical wire *switches* from a red into a blue wire it uses a grid point of an interleaved track as a contact point.

The algorithm determines the wiring column by column, from left to right. It tries to run all horizontal wires as red wires and all vertical wires as blue wires. This is not always possible: when a net reaches its final column as a red wire, the wire immediately continuing on this track has to start off as a blue wire. The algorithm removes such a blue wire from the track within the next two columns by continuing it on another track as either a red wire, or as a blue wire (if it again participates in a knock-knee). We will show that at most 3 blue horizontal wires can be between any two columns in the channel. Furthermore, a horizontal red wire changes into a blue wire only when its corresponding net reaches the final column (in some cases it will be the column before the final one).

In the j-th step of the algorithm we determine the vertical wires in column j, and the horizontal wires between column j and column $j+1$. All horizontal red wires - except the one corresponding to the net ending in column j, and possibly the one ending in column $j+1$ - continue as red wires. We thus only consider the horizontal blue wires, which we try to change into red wires, and the net starting in column j, the net ending in column j, and, in some cases, the net ending in column $j+1$. We first describe the two routines used when processing one column. The slipping routine, which changes blue wires into red wires by letting them 'slip' onto other tracks, and the take-down routine, which runs vertical wires from one track to another track and uses contact points to cross over blue horizontal wires.

The *slipping routine* has one argument i, a track number, and causes each blue wire on track 1 to $i-1$ to 'slip' onto a higher numbered track (less than i) and change into a red wire. Thus, after the the slipping routine tracks 1 to $i-1$ are guaranteed to contain only red wires, and track i will contain a red or a blue wire.

Let $b_1, b_2, \ldots, b_c$ be the tracks containing a blue wire between column $j-1$ and j, $b_l < b_{l+1}$, $1 \le l < c$, $b_c \le i-1$, and let n_j be the net starting in column j. Then net n_j runs down column j as a blue wire, switches into a red wire between track b_1-1 and b_1, and continues as a red wire on track b_1. The blue wire in track b_l behaves the same way: it runs down column j as a blue wire and switches into a red wire between track $b_{l+1}-1$ and b_{l+1}, $l<c$. The wire on track b_c runs down column j as a blue wire and continues on track i as either a blue or a red wire, depending on the color of the wire on track i between column $j-1$ and j. See Fig. 2.1.

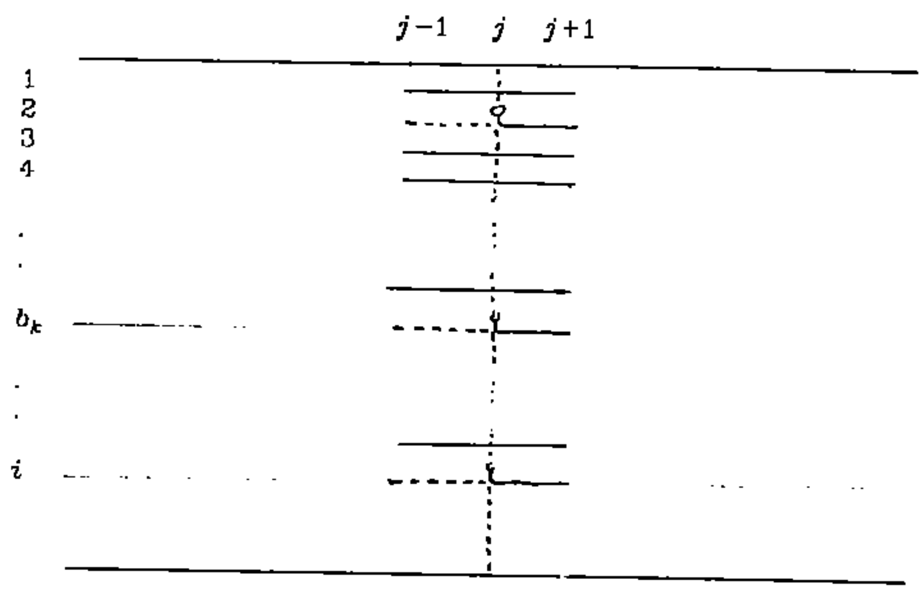


Fig. 2.1 Slipping Routine
(Red wires are denoted by solid lines, blue wires by dashed lines)

The number of contact points needed in the slipping process is equal to the number of blue wires on track 1 to i. The time required by the slipping routine is proportional to the number of blue wires between column $j-1$ and j.

The *take-down routine* has two arguments, i_1 and i_2, both of which are track numbers, $i_1 \le i_2$. We run the wire currently on track i_1 down column j until it reaches track i_2 and use contact points as needed. If $i_2 = d+1$, the wire ends at the terminal in column j; in all other cases it continues on track i_2 (its color is determined by the wire previously on track i_2). See Fig. 2.2.

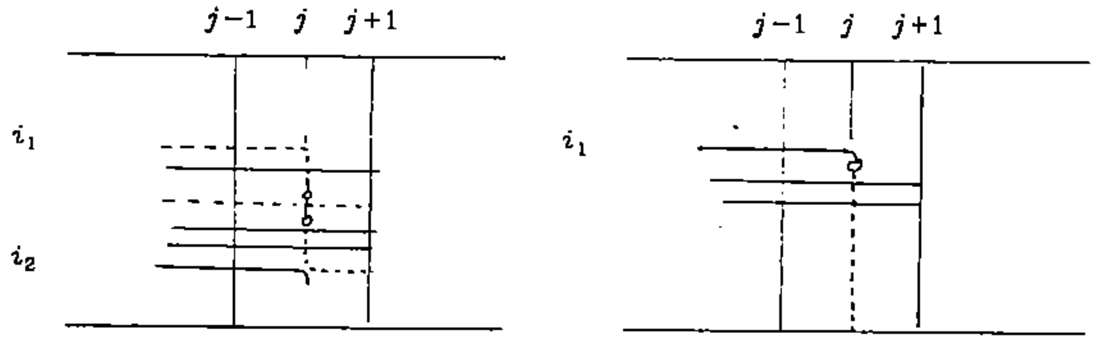
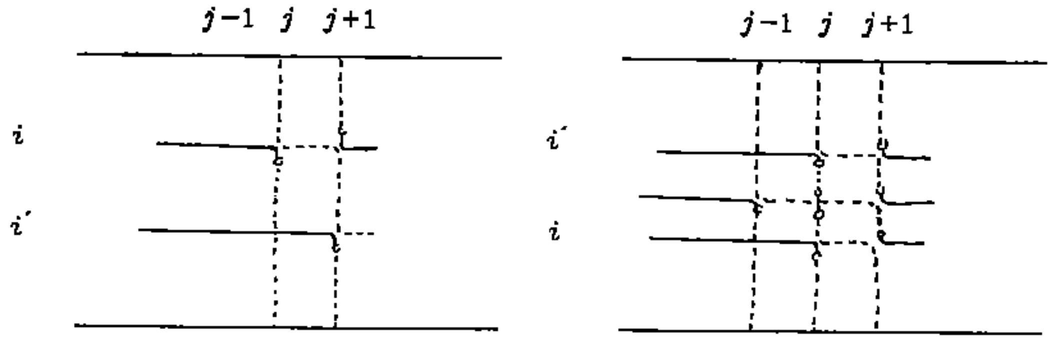


Fig. 2.2: Take-down Routine

Assume that net n_l ending in column j is currently on track i, and that net n_j starts in column j. If the wire on track i belonging to net n_l is *blue*, we perform the slipping routine down to track i and the take-down routine from track i to $d+1$. See Fig. 2.1. In column $j+1$ we will have only red wires between track 1 and i (a more careful analysis of the algorithm shows that actually *all* wires are red in column $j+1$).

If the wire on track i belonging to net n_l is *red*, we are forced to put a blue wire on track i between column j and $j+1$. In order to have a red wire on track i and to continue the blue horizontal wire currently on track i as a horizontal red wire as soon as possible, we look at the net ending in column $j+1$. We distinguish between two cases. First assume that the net ending in column $j+1$ is on a track below track i. When the blue wire now on track i participates in the slipping routine for the net ending in column $j+1$, a red wire is put on track i in column $j+1$. The blue wire slips from track i onto a higher numbered track, where it could still run as a blue wire. See Fig. 2.3 (a).



Net ending in column $j+1$ is
(a) below track i (b) above track i

Fig. 2.3

In the second case the net ending in column $j+1$ is on track i', which is above track i. We perform the slipping routine down to track i', take the wire previously on track i' down to track i, and take the red wire on track i belonging to net n_i from track i down to $d+1$. See Fig. 2.3 (b). Track i contains now a blue wire that ends in column $j+1$. Thus all the blue horizontal wires between track i' and track i (which were crossed over in take-down(i',i) will be changed into red wires in the next step. In the worst case we have 3 blue horizontal wires between two columns. This occurs when a red wire ends in column $j-1$ and the first case holds, and a red wire ends in column j and the second case holds, as shown in Fig. 2.3 (b). (Note that between column $j+1$ and $j+2$ we will then only have red wires.)

We now give the detailed algorithm for the wiring of the d runs. For simplicity we introduce a 'null' net for each run; this allows us to start off with all tracks containing a red wire between column 0 and column 1. Let (p_1^k, q_1^k) be the first net of each run, and order the runs such that $p_1^k < p_1^{k+1}$. The column containing p_1^1 will be column 1. Let $(-k+1, p_1^k)$ be the null net for run k, $1 \le k \le d$. The horizontal part of this net is put as a red wire onto track k, all the vertical parts of the net are run as blue wires.

Wiring Algorithm

```
1. initialize the channel with the d null nets;
   j := 1

2. while not all nets have been wired do
   (* the net ending in column j is on track i *)
   (* color(i,j) gives the color of the wire on track i between column j-1 and j *)

       i := track number of net ending in column j;
       if color(i,j) = blue
         then begin (* an ending blue wire removes all blue wires above track i *)
                slip(i);
                take-down(i,d+1)
                end
         else begin   (* an ending red wire causes a blue wire on track i *)
                i' := track number of net ending in column j+1;
                if i' ≥ i
                  then slip(i)
                  else begin  (* make sure the blue wire disappears soon *)
                         slip (i');
                         take-down(i',i)
                         end
                take-down(i,d+1);
                end
     j := j+1
   endwhile
```

The wiring algorithm produces nets of the shapes shown in Fig. 2.4. Each wired net consists of 3 parts: the *slipping part*, which consists only of blue wires and contains l *slips*, $0 \le l \le d$. The first $l-1$ slips can only occur when the blue horizontal wire is the last wire participating in the slipping routine and the net ending in the corresponding column is red. The horizontal blue wire following the last slip can be of length 2 (when the situation described in Fig. 2.3 (b) occurs), all other horizontal blue wires have length 1. The *red part* of the wired net consists of a horizontal red wire ending in the column in which the net ends (Fig. 2.4 (a)), or one column before (Fig. 2.4 (b) and (c)). The *take-down* part the wired net consists either of a straight blue wire, or a blue wire with one slip and possibly two contact points before the slip. Thus, each wired net contains at most 4 contact points, and the total number of contact points is at most $4n$.

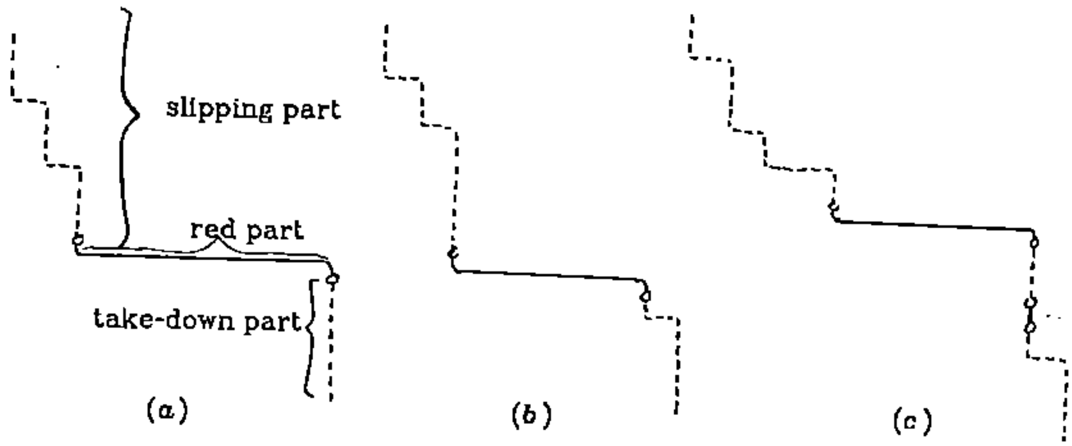


Fig. 2.4: Shapes of the wired nets

Since we only have to consider the nets ending in column j (and possibly $j+1$), the net starting in column j, and the blue wires entering column j, one column is processed in constant time. Hence, the Wiring Algorithm wires a right CRP in time $O(n)$ and uses $O(n)$ contact points.

The number of slips in the slipping part of a wired net can be bounded for the cost of two additional contact points (they will occur in the first vertical wire of the slipping part of a net). Assume no more than s slips are wanted in a net, $s \geq 1$. Let the net ending in column j be on track i, and let w be the wire that would have to slip again in slip(i). Instead of making a slip in wire w, run w horizontally and let the net starting in column j slip onto track i. See Fig. 2.5. Wire w will then change into a red wire in column $j+1$ or $j+2$.

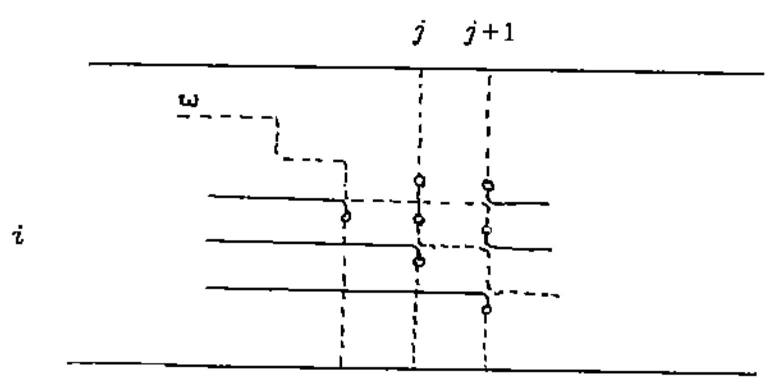


Fig. 2.5: Bounding the number of slips

3. Conclusions

The algorithm from section 2 can be modified to solve *weak* CRPs with $2d-1$ tracks and $O(n)$ contact points. A weak CRP is a CRP that contains right and left nets with the following restriction. For all columns p in which one terminal belongs to a right net and the other terminal to a left net, we either have the right net of the form (p,q) and the left net of the form (r,p), or we have the right net of the form (q,p) and the left net of the form (p,r), $r \neq q$. Thus, the situations shown in Fig. 3.1 (a) and (b) cannot occur together in a weak CRP.

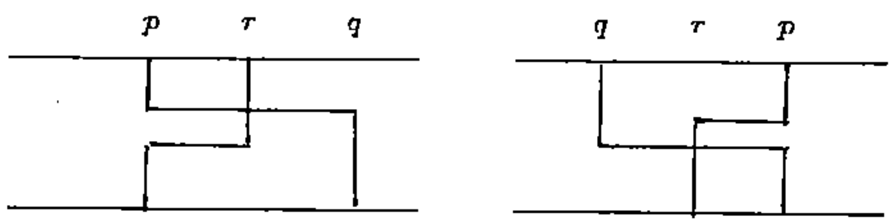
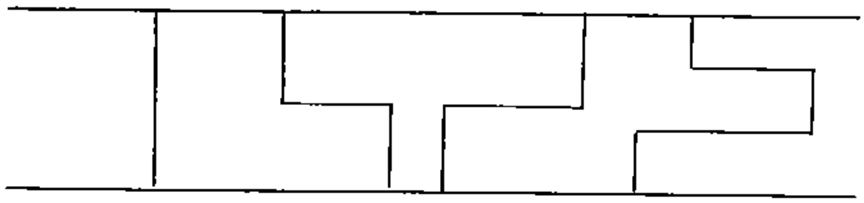


Fig. 3.1: (a) (b)

Weak CRPs have a wire layout that consists only of simple shaped nets. If situation 3.1 (a) occurs the algorithm processes the channel left to right; if situation 3.1 (b) occurs right to left. During the algorithm right nets and left nets are handled separately; i.e., a vertical wire of a right net passes over a horizontal blue wire of a left net and vice-versa. Hence the algorithm contains a right and a left slipping routine and the number of contact points needed is still $O(n)$.

If we remove the restriction of Fig. 3.1 the idea used in the previous algorithms fails. First, the d-track wire layouts for general CRPs have to contain some nets of non-simple shape. Preparata and Lipski show that nets of the shapes shown in Fig. 3.2 are sufficient, [PL]. For nets of the form (p,q) and (q,p) the shape shown in Fig. 3.2 (d) is necessary, and for some CRPs using only the shapes shown in Fig. 3.2 (a) - (c) results in $\Omega(n)$ tracks. But the main problem is that the ordering of the wires in the channel is now crucial, and the use of a slipping routine to remove as many blue horizontal wires as possible destroys the ordering.



(a) (b) (c) (d)
Fig. 3.2: Shape of the nets in the wire layout of [PL]

At the present, it is not known whether $2d-1$ tracks and $O(n)$ contact points can be achieved simultaneously. A proof that it cannot be achieved for all CRPs might use a 'worst-case' CRP that contains nets of the form shown in Fig. 3.1, and nets that require the shape shown in Fig. 3.2 (d) in d-track wire layout. Another interesting question is how many tracks are needed to achieve $O(n)$ contact points. The knock-knees force the horizontal wires to run - for at least a short distance - on the 'bad' layer, but avoiding knock-knees altogether requires $n^{1/2}$ tracks for some CRPs, [BR]. It is quite possible that the problem does not get easier if cd, $c>2$ tracks can be used.

References

[BB] T. Bolognesi, D.J. Brown, 'A Channel Routing Algorithm with Bounded Wire Length', unpublished manuscript, 1982.

[BP] D.J. Brown, F.P. Preparata, 'Three-Layer Channel Routing of Multiterminal Nets', draft, 1983.

[BR] D.J. Brown, R.L. Rivest, 'New Lower Bounds for Channel Width', Proceedings of the CMU Conf. on VLSI Systems and Computations, pp 153-159, 1981.

[D] D.N. Deutsch, 'A Dogleg Channel Router', Proceedings of the 13th IEEE Design Automation Conf., pp 425-433, 1976.

[DKSSU] D. Dolev, K. Karplus, A. Siegel, A. Strong, J.D. Ullman, 'Optimal Wiring between Rectangles', Proceedings of the 13th Annual ACM Symp. on Theory of Computing, pp 312-317, 1981.

[L] F.T. Leighton, 'New Lower Bounds for Channel Routing', unpublished manuscript, 1981.

[PL] F.P. Preparata, W. Lipski, 'Three Layers are enough', Proceedings of the 23rd Annual IEEE Foundations of Comp. Sc. Conf., pp 350-357, 1982.

[RBM] R.L. Rivest, A.E. Baratz, G. Miller, 'Provably Good Channel Routing Algorithms', proceedings of the CMU Conf. on VLSI Systems and Computations, pp 153-159, 1981.