Local Routing of Two - Terminal Nets is Easy

(Extended Abstract)

A 84/12

by
Michael Kaufmann
Kurt Mehlhorn

FB 10
Universität des Saarlandes
6600 Saarbrücken
West-Germany

November 1984

Abstract: A local routing problem is given by a routing region (a subgraph of the planar grid) and a set of nets. For each net a global routing is also given. The problem is to find a local routing which is consistent with the global routing (if there is one). In this paper we show that the local routing problems can be solved in time $O(n\ (\log n)^2)$.

1. Introduction

Automatic design systems, e.g. CALCOS (Lauther) and PI (Rivest), for integrated circuits divide the routing problem into several stages.

1) Determine a global routing for every net. A net is a set of points which have to be connected and a global routing fixes the global shape of the realization of the net, i.e. how the net runs with respect to the subcircuits (cf. Figures 1 and 2).

2) Cut the routing region into regions of simple shape, e.g. channels.

3) Determine for every net the exact positions where it crosses channel boundaries.

4) Route each channel.

In some systems, e.g. CALCOS, stages 3 and 4 are combined into a single stage. Channels are routed one by one and the routings in the first i channels fix the positions of nets which leave these channels. In all stages heuristic algorithms are usually used. <u>In this paper we show that stages 2) to 4) can be combined to a single stage (called the local routing problem (LRP)) which can be solved efficiently.</u> More precisely, we will show that LRP's for two-terminal nets can be solved in time $O(n(\log n)^2)$ where n is the size of the routing region and that LRP's for multi-terminal nets can be solved approximately in that time bound. An exact statement is given in section 2.

Previous theoretical work on routing concentrated on routing regions with "simple" shapes. Very good routers for channels (Rivest/ Barratz/ Miller, Preparata/ Lipski, Baker/ Bhatt/ Leighton), switchboxes (Mehlhorn/ Preparata, Frank) and generalized switchboxes (Kaufmann/ Mehlhorn) were found. Routing in general planar graphs was considered by Okamura/

Seymour and Becker/ Mehlhorn. Note however that all papers mentioned require the terminals of all nets to be on the boundary of the <u>same</u> face of the routing region. A notable exception is the paper by Baker/ Pinter who considered <u>river routing</u> within a ring of pads. We should finally mention that the combination of stages 1) to 4) is NP-complete (Kramer/ v. Leeuwen).

This paper is organized as follows. In section 2 we give a precise definition of the LRP for two-terminal nets, the multi-terminal net case can be found in section 6. Section 3 gives the algorithm, in section 4 we prove it correct and in section 5 we describe an implementation and analyse its running time.

2. Precise Problem Definition

The <u>planar rectangular grid</u> consists of vertices $\{(x,y)\ ;\ x,y \in \mathbb{Z}\}$ and edges $\{((x,y),\ (x',y'))\ ;\ |x-x'| + |y-y'| = 1\}$. A <u>routing region</u> R is a finite subgraph of the planar rectangular grid. A routing region is a <u>full polygon</u> if every finite face of R has exactly four vertices.

In the sequel R always denotes a routing region. Let M be the set of finite faces of R which have five or more boundary vertices and let $\overline{M}$ be M together with the infinite face. Let B be the set of vertices of R with degree at most three. Note that a vertex $v \in B$ lies on the boundary of a face $F \in \overline{M}$.

A <u>local routing</u> is a path in the routing region R. Two local routings p and q are elementarily equivalent if there are paths p_1,p_2,q_2,p_3 such that $p = p_1p_2p_3$, $q = p_1q_2p_3$ and $p_2q_2^{-1}$ (q_2^{-1} is the reverse of path q_2) is the boundary cycle of a face $F \notin \overline{M}$. Two local routings p and q are <u>equivalent</u> if there is a sequence $p_o,\ldots,p_k$, $k \geq 0$, of paths such that $p = p_o$, $q = p_k$ and p_i and p_{i+1} are elementarily equivalent for $0 \leq i < k$. Note that if p and q are equivalent then p and q have the same endpoints and the cycle pq^{-1} does not enclose a face $F \in M$ (cf. Figure 1).

We use [p] to denote the equivalence class of local routing p. A global routing is an equivalence class [p] . A net N is a triple (s,t,gr) where s and t are vertices in B and gr is a global routing connecting s and t, i.e. gr = [p] where p is a path from s to t. We call gr = gr(N) the global routing of net N and s and t its terminals.

We are now ready to state the Local Routing Problem (LRP).

Input: A routing region R and a set Ne of nets.

Output: A local routing $\ell r(N)$ for each net $N \in Ne$ such that

1) $\ell r(N) \in gr(N)$ for all N
2) $\ell r(N_1)$ and $\ell r(N_2)$ are edge-disjoint for $N_1, N_2 \in Ne$, $N_1 \neq N_2$

or an indication that there is no such set of local routings.

Figure 2 gives an example. In this paper we will prove the following theorem.

Theorem 1: Let P = (R,Ne) be an even LRP where R has n vertices. In time $O(n(\log n)^2)$ one can decide whether P has a solution and also construct a solution if it does.

It remains to define even LRP. The multiple source dual D(R) is defined as follows. (cf. Figure 3).
For every edge e of R there is a dual edge d(e) with its endpoints lying in those faces of R which are separated by e. The endpoints of edges which lie in faces outside $\overline{M}$ are identified, the endpoints in faces in $\overline{M}$ are kept distinct and are called sources of the dual graph. A cut of R is a simple path in the dual graph connecting two sources. The capacity cap(C) of a cut C is its lengths (= number of edges of R intersected by the cut) (cf. Figure 3). A cut can be viewed as a polygonal line $s_1, \ldots, s_k$ where each s_i is a straight-line segment and s_i and s_{i+1} have a different direction (one horizontal, one vertical). A cut is simple if $k \leq 2$, i.e. if the cut has at most one bend.

Let C be a cut and p be a local routing. Then $cross(p,C)$ is the number of edges e of p with $d(e)$ in C, i.e. the number of times p goes across C. For a global routing gr we define

$$cross(gr,C) = \min \{cross(p,C) \; ; \; p \in gr\}.$$

Finally, the <u>density</u> $dens(C)$ of cut C is defined by

$$dens(C) = \Sigma \{cross(gr(N), C) \; ; \; N \in Ne\}$$

and the <u>free capacity</u> $fcap(C)$ is given by

$$fcap(C) = cap(C) - dens(C)$$

A cut C is saturated if $fcap(C) = 0$ and oversaturated if $fcap(C) < 0$.

An LRP is <u>even</u> iff $fcap(C)$ is even for every cut C.

We show (as part of the proof of theorem 1)

<u>Theorem 2:</u> If $fcap(C)$ is nonnegative and even for every simple cut C then the routing problem P has a solution. If P has a solution then $fcap(C)$ is nonnegative for every cut C.

For the sequel the following alternative definition of $cross(gr,C)$ is useful. Local routing $p \in gr$ is <u>reduced</u> with respect to C if p cannot be written $p = p_1 e_1 p_2 e_2 p_3$ where $d(e_1)$, $d(e_2) \in C$ and there is a path p_4 such that $e_1 p_2 e_2$ and p_4 are equivalent and $cross(p_4,C) = 0$.

<u>Lemma 1:</u> If local routing p is reduced with respect to C then $cross(p,C) = cross([p],C)$.

We infer from lemma 1 that we can use reduced local routings to count crossings of cuts with global routings at least as far as one cut is concerned. We will now extend this observation to several cuts.

Let C and D be cuts. C and D are <u>interferencefree</u> if they are either vertex-disjoint or if $C = EC'$, $D = ED'$ and C' and D' are vertex-disjoint except for their common start vertex (cf. Figure 4). A set S of cuts is <u>interferencefree</u> if the cuts in S are pairwise interferencefree. A local routing p is <u>reduced with respect to</u> S if it is reduced with respect to all cuts in S.

Lemma 2: Let S be an interferencefree set of cuts and let gr be a global routing. Then there is a $p \in gr$ such that p is reduced with respect to S.

3. The LRP Algorithm

In section 3.1 we give our LRP algorithm. In its description we use two undefined concepts: minimal saturated cut and leftmost decomposition of a net.
The (very lengthy) definition of these concepts is given in section 3.2. In section 3.2 we will also prove several facts about these concepts.

3.1 The Algorithm

The algorithm processes the routing region row by row from top to bottom. Within each row we proceed from left to right. A vertex $a = (x,y)$ of the routing region R is called the left upper corner of R if there is no vertex (x',y') of R with $y' > y$ or $y' = y$ and $x' < x$. We use b to denote the vertex $b = (x,y-1)$ and $e = (a,b)$ to denote the vertical edge incident to a.

The algorithm constructs a solution iteratively. In each iteration it first simplifies the routing region such that all cuts have capacity two or more and then it considers edge e and decides whether to route a net across e or not. In either case edge e is deleted from the routing region. Thus $O(n)$ iterations suffice. The details are as follows. In the description of the algorithm we assume that we start with a solvable LRP. If the algorithm is applied to a nonsolvable LRP then it will find an oversaturated cut at some point.

```
(1)  Simplify;
(2)  while  E ≠ ∅
(3)  do   - - fcap(Y) is even and nonnegative for every simple
          - - cut Y. Every cut of routing region R has
          - - capacity at least two;
(4)            let a be the left upper corner of R and let e = (a,b)
               be the vertical edge incident to a;
(5)            if  there is a saturated simple cut through edge e
(6)            then  let X be the minimal saturated simple cut
                     through edge e;
(7)                  Among the nets which go across cut X let N be
                     the one with the leftmost decomposition (N1,N2)
                     where N1 = (s,a,gr1) and N2 = (b,t,gr2);
(8)                  delete edge e and net N and add nets N1 and N2
                     - -  the local routing of net N consists of
                     - -  the local routings for nets N1 and N2
                     - -  and edge e.
               else  - -  a is not terminal of any net
(9)                  delete all four edges of the finite face on whose
                     boundary a and b lie (cf. Figure 5)
               fi ;
(10)           simplify
     od
```

where procedure Simplify is given by

```
proc  Simplify
while   ∃ cut X with cap(X) = 1
do      let e be the edge cut by X and let N be the net with
        cross(N,X) = 1. Let (N1,N2) be a decomposition of N
        with respect to X;
        delete edge and net N and add nets N1 and N2
        - -  the local routing for N consists of the local
        - -  routings for N1 and N2 and edge e
od
```

3.2 The Missing Concepts

In this section we define the concepts of minimal saturated simple cut and leftmost decomposition of a net.

Throughout this section we assume that there are no cuts of capacity one.

We first define minimal saturated simple cut. A simple cut through edge e consists of a horizontal segment s_1 and maybe a vertical segment s_2. Segment s_2 either bends upwards or downwards. We postulate that all cuts with s_2 bending upwards are smaller than the cut with no bend which in turn is smaller than all cuts with s_2 bending downwards.
We furthermore order the cuts with s_2 bending upwards according to increasing length of s_1, i.e. the shorter s_1 the smaller the cut, and the cuts with s_1 bending downwards according to decreasing length of s_1, i.e. the longer s_1 the smaller the cut (cf. Figure 6).

We will next prove some important facts about saturated cuts.

<u>Lemma 3:</u> Let P be a local routing problem.
a) If there is an oversaturated cut then there is a simple oversaturated cut.
b) If $Y = s_1, s_2, \ldots, s_k$, $k \geq 3$, (the s_i are straight-line segments) is a saturated cut through edge e then there either exists an oversaturated simple cut (not necessarily through e) or there exists a saturated simple cut X through e which is smaller than the cut s_1, s_2' where s_2' extends s_2 until it intersects a boundary edge. (cf. Figure 7).

We can now turn to the definition of leftmost decomposition. We do so in a three step process. We first define <u>slicings</u> of the routing region. We then use slicings to define the ordering <u>left-of</u> on nets with a common endpoint, and then to extend this ordering to decompositions of nets. We will also prove that the ordering is independent of the particular slicing used in its definition.

A <u>slicing</u> S of R is a set $C(F)$, $F \in M$, of cuts and a function parent: $M \rightarrow \bar{M}$ such that

1) $C(F)$ has one endpoint in face F and the other endpoint in face parent(F) $\in \bar{M}$

2) the function parent defines a tree on $\bar{M}$ with the infinite face beeing the root

3) cuts $C(F)$ and $C(G)$ do not interfere if $F \neq G$.

Figure 8 illustrates this definition. If S is a slicing then removal of the edges e intersected by a cut in S turns R into a full polygon which we denote P(S).

We will next define a cyclic ordering on the set $B \cup C$ where B is the set of vertices of R of degree three or less and $C = \{C(F)^{+1}, C(F)^{-1}; F \in M\}$. Here, $C(F)^{+1}$ and $C(F)^{-1}$ are new symbols which represent the two sides of cut C(F). The ordering on $B \cup C$ is defined by a counterclockwise traversal of the full polygon P(S) where $C(F)^{+1}$ represents the sequence of vertices used to rèach face $F \in M$ from face parent(F) and $C(F)^{-1}$ represents the path back to parent(F).

<u>Remark:</u> This ordering is well-defined since no node in B can belong to the boundary of two faces in $\overline{M}$. Otherwise, there would be a cut of capacity one.

We can use slicings to decompose nets into <u>elementary pieces</u>. Let N = (s,t,[p]) be a net where p is reduced with respect to S. The net N crosses some cuts in S ; each cut C(F) is crossed either in the direction from $C(F)^{+1}$ to $C(F)^{-1}$ or from $C(F)^{-1}$ to $C(F)^{+1}$. In this way we can associate with the net an element in $B(C^2)^* B$.

$$s\ C(F_1)^{+d_1}\ C(F_1)^{-d_1}\ C(F_2)^{+d_2}\ C(F_2)^{-d_2} \ldots C(F_k)^{-d_k}\ t$$

where $d_i \in \{+1, -1\}$ and N crosses $C(F_i)$ in the direction from $C(F_i)^{d_i}$ to $C(F)^{-d_i}$. The <u>elementary pieces</u> of net N are now given by $(s, C(F_1)^{d_1})$, $(C(F_1)^{-d_1}, C(F_2)^{d_2})$, ... An elementary piece is an element of $(B \cup C)^2$.

We are now ready to define the ordering <u>left of</u> on nets with a common terminal. Let $N_i = (s,t_i,[p_i])$ be nets and let $ep_{i1},\ldots, ep_{ik_i}$ be the decomposition of N_i into elementary pieces, i = 1,2. Then N_1 is left of N_2 iff there is a j such that $ep_{1\ell} = ep_{2\ell}$ for $\ell <$ j , $ep_{1,j} = (u,v)$, $ep_{2,j} = (u,w)$ with $u,v,w \in B \cup C$, $v \neq w$ and u, v, and w occuring in <u>that</u> order in the cyclic ordering of $B \cup C$. (cf. Figure 9).

The ordering left of is defined with respect to a particular slicing S. It is however independent of the slicing (cf. Figure 10).

Lemma 4: The ordering left of does not depend on the slicing used in its definition.

We can now finally define decompositions of nets and the ordering left of on decompositions. Let X be a cut through edge $e = (a,b)$. A pair $N_1 = (s,a,[p_1])$, $N_2 = (b,t,[p_2])$ of nets is a decomposition of net $N = (s,t,[p])$ (with respect to cut X) if $[p] = [p_1\ e\ p_2]$ and cross (N_1,X) + cross (N_2,X) = cross $(N,X) - 1$. Decomposition (N_1,N_2) of net N is left of decomposition (M_1,M_2) of net M if N_2 is left of M_2.

4. Correctness

In this section we prove the correctness of our algorithm. Let P be an even solvable LRP. We establish the following invariant for our algorithm.

(Invariant): fcap(Y) is even and nonnegative for all simple cuts Y.

The invariant is certainly true initially; fcap(Y) is even because P is even and fcap(Y) is nonnegative because P is solvable. It remains to show that the invariant is maintained.

Lemma 5: An application of procedure Simplify maintains the invariant.

Lemma 6: An execution of the main-loop maintains the invariant.

Proof: The proof is very lengthy and requires a detailed case analysis. We sketch one case. Assume that there is a saturated cut through edge e; let X be the minimal saturated simple cut through e. Let Y be any simple cut. We have to show that fcap'(Y) is even and nonnegative in the modified problem (indicated by the prime). We discuss the case that X and Y cross and have exactly one common vertex v (cf. Figure 11). Assume that v is closer to e than e'. Then fcap'(Y) = fcap(Y) - 2. Thus the only case to discuss is fcap(Y) = 0. We consider cuts Z_1 and Z_2 as shown in Figure 12 and show fcap(Z_1) + fcap(Z_2) = fcap(X) + fcap(Y) = 0. Hence either fcap(Z_1) = 0, a contradiction to the minimality of cut X or fcap(Z_2) < 0, a contradiction to the invariant (with a tacit application of lemma 3). In either case we have derived a contradiction and hence fcap(Y) > 0.

It remains to show fcap(Z_1) + fcap(Z_2) = fcap(X) + fcap(Y). Note first that cap(Z_1) + cap(Z_2) = cap(X) + cap(Y). We consider the densities next. Let S be a slicing such that S $\cup$ {X} and S $\cup$ {Y} are interferencefree. The cuts X and Y divide B $\cup$ C into four intervals, say G, H, I, K, as indicated in Figure 12. For subsets U,V of B $\cup$ C let $\overline{\text{dens}}(U,V) = |\{ep ; ep = (u,v)$ is an elementary piece with $u \in U, v \in V\}|$.
We have dens(X) = $\overline{\text{dens}}(G\cup H, I\cup K)$, dens(Y) = $\overline{\text{dens}}(G\cup K, H\cup I)$, dens($Z_1$) = $\overline{\text{dens}}(K, G\cup H\cup I)$, dens($Z_2$) = $\overline{\text{dens}}(H, I\cup K\cup G)$ and therefore dens(Z_1) + dens(Z_2) = dens(X) + dens(Y) - 2 $\overline{\text{dens}}(I,G)$. Next note that $\overline{\text{dens}}(I,G) = 0$. This can be seen as follows. Assume otherwise. Then there is an elementary piece ep'$\in$ I $\times$ G. Let N with decomposition (N_1,N_2) be the net chosen by the algorithm to be routed across edge e. Let ep be the elementary piece of N which "contains" edge e'. Then ep $\in$ I $\times$ H and hence (N_1,N_2) is not a leftmost decomposition. This contradiction shows that $\overline{\text{dens}}(I,G) = 0$ and hence dens(Z_1) + dens(Z_2) = dens(X) + dens(Y). □

Theorem 2 now follows immediately from lemma 6.

5. Implementation and Running Time

There are two main ingredients to the implementation. The first idea is to represent nets as sequences of elementary pieces, to represent pieces as pairs of integers, and to store these pairs in a range tree (cf. Mehlhorn, section VII.2.2).

These integers are obtained as follows. We number $B \cup C$ in counterclockwise order where we use a single integer for an element in B and use an interval of length cap(C(F)) for the elements $C(F)^{+1}$, $C(F)^{-1}$ in C. In a sense we number the endpoints of the edges going across cut C(F); cf. Figure 13.
Figure 13 also illustrates how pairs of integers are assigned to elementary pieces.

Lemma 7: The pairs representing the elementary pieces can be computed in time $O(n(\log n)^2)$

From now on we are on safe grounds and can essentially use the methods developed in Kaufmann/Mehlhorn.

We will next show how the range tree can be used to find the net to be routed across edge e in line (7) of the algorithm. Let X be a cut through edge e. Cut X partitions $B \cup C$ into two sets L and R with $b \in L$, $a \in R$. In our numbering of $B \cup C$ the set L corresponds either to an interval $[j,h]$ with $j < h$ or to a pair of intervals $[j,r],[1,h]$ with $j > h$ and r the largest number used in the numbering. In the former case (the latter case being similar) the net N to be routed across edge e is characterized by the elemtary piece ep with representation (f,g), $f,g \in \mathbb{N}$ where $f \notin [j,h]$, $g \in [j,h]$ and g minimal. This pair can be found in time $O(n(\log n)^2)$ using the range tree.

The second idea is to use an auxiliary data structure for the top row of the routing region which contains the free capacities of all cuts going through edge e. This data structure is a priority queue with updates as described in Galil/Naamad, see also Mehlhorn, section IV.9.1 . If the top row has length L then this data structure can be constructed in time $O(L(\log n)^2)$ using the range tree. It can be used to decide in time $O(\log n)$ whether there is a saturated cut through edge e (line 5) and to find the minimal such cut (line 6). Also it can be updated in time $O(\log n)$ per iteration of the main loop.

Altogether we obtain an $O(n(\log n)^2)$ algorithm.

6. Multi-terminal Nets

A local routing for a multi-terminal net is a tree, a global routing is an equivalence class of trees. Density and free capacity are defined as in the two-terminal case. We have:

Theorem 3: Let P be an LRP with multi-terminal nets. If $2\ \mathrm{dens}(X) < \mathrm{cap}(X)$ for every cut X then P has a solution and this solution can be constructed in time $O(n(\log n)^2)$.

7. References

1. B.S. Baker/ S.N. Bhatt, F.T. Leighton "An Approximation Algorithm for Manhattan Routing
   15th STOC 1983, 477-486

2. B.S. Baker/ R.Y. Pinter "An Algorithm for the Optimal Placement and Routing of a Circuit within a Ring of Pads,
   24th FOCS, 1983, 360-376

3. M. Becker/ K. Mehlhorn "Algorithms for Routing in Planar Graphs"
   Technical Report, FB 10, Univers. des Saarlandes
   August 1984

4. A. Frank "Disjoint Paths in Rectilinear Grids"
   Combinatorica 2,4 (1982), 361-371

5. Z. Galil/ A. Naamad "Network Flow and Generalized Path Compressions"
   11th STOC 1979, 13-26

6. M. Kaufmann/ K. Mehlhorn "Generalized Switchbox Routing"
   Technical Report, FB 10, Univers. des Saarlandes, 1984

7. M.E. Kramer/ J. van Leeuwen "Wire Routing is NP-complete"
Technical Report RUU-CS-82-4, 1982, Utrecht

8. U. Lauther "The SIEMENS CALCOS System for Computer Aided Design of Cell Based IC Layout"
Proc. 1st ICCC, 1980, 768-771

9. K. Mehlhorn "Data Structures and Efficient Algorithms"
Vol. 1: Sorting and Searching
Vol. 2: Graph Algorithms and NP-completeness
Vol. 3: Multi-dimensional Data Structures and Computational Geometry
Springer Verlag, EATCS Monographs in Theoretical Computer Science, 1984

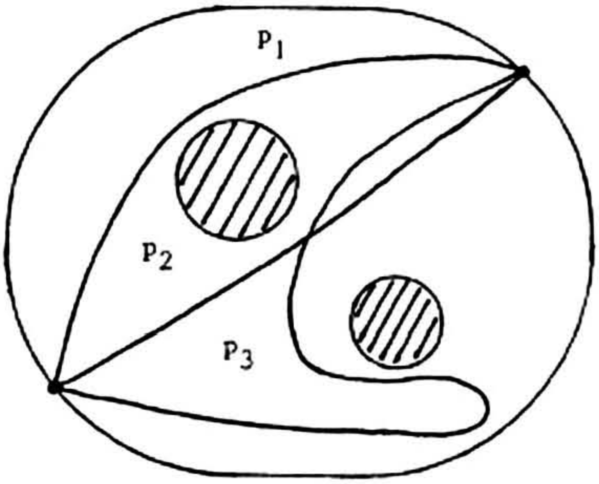
10. K. Mehlhorn/ F. Preparata "Routing Through a Rectangle"
Technical Report, 1983, to appear in JACM

11. H. Okamura/ P.D. Seymour "Multicommodity flows in planar graphs"
Journal of Combinatorial Theory, Series B, 1981, 75-81

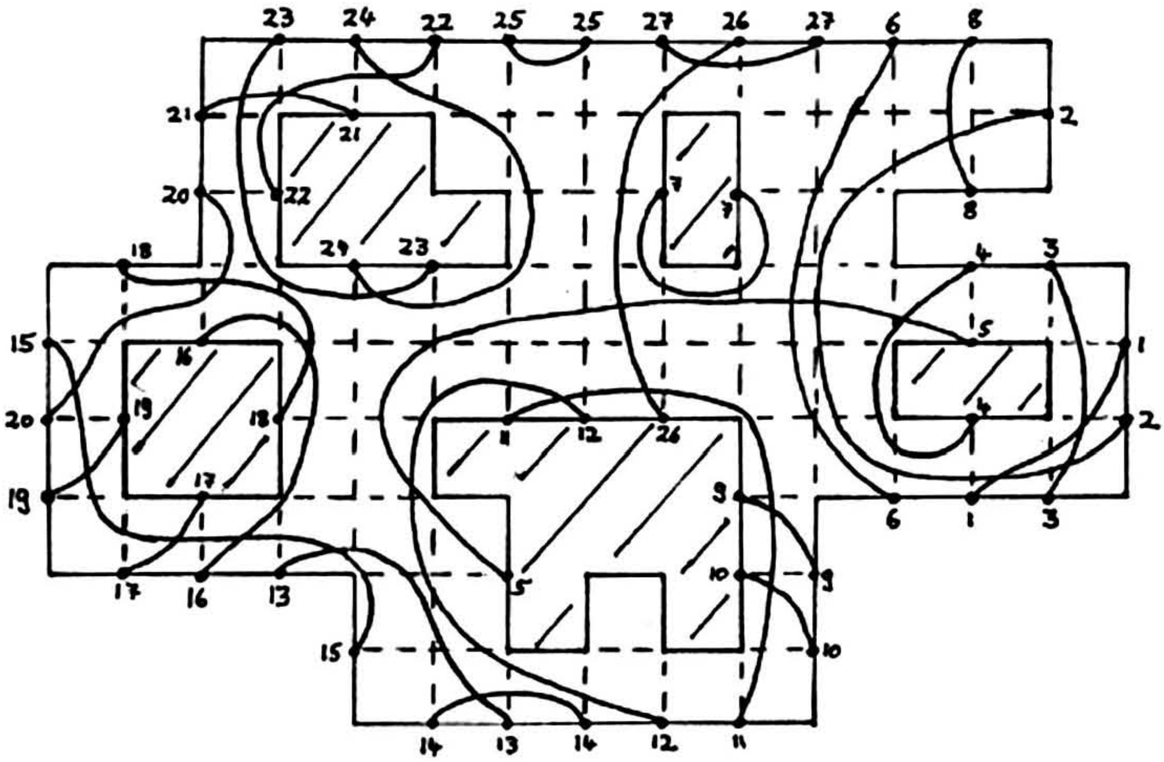
12. F. Preparata/ W. Lipski "Three Layers are Enough"
23rd FOCS 1982, 350-357

13. R.L. Rivest "The PI (Placement and Interconnect) System"
Proc. of the 19th Design Automation Conference, 475-481, 1982

14. R.L. Rivest/ A.E. Barratz/ G. Miller "Provably Good Channel Routing Algorithms"
Proc. of the CMU Conf. on VLSI Systems and Computations, 153-159, 1984



<u>Figure 1:</u> Subcircuits (Faces F in M) are hatched. p_1 and p_2 are not equivalent, p_2 and p_3 are equivalent.



<u>Figure 2a):</u> A local routing problem. Nets and their global routings are indicated as "rubber bands". Subcircuits are hatched.

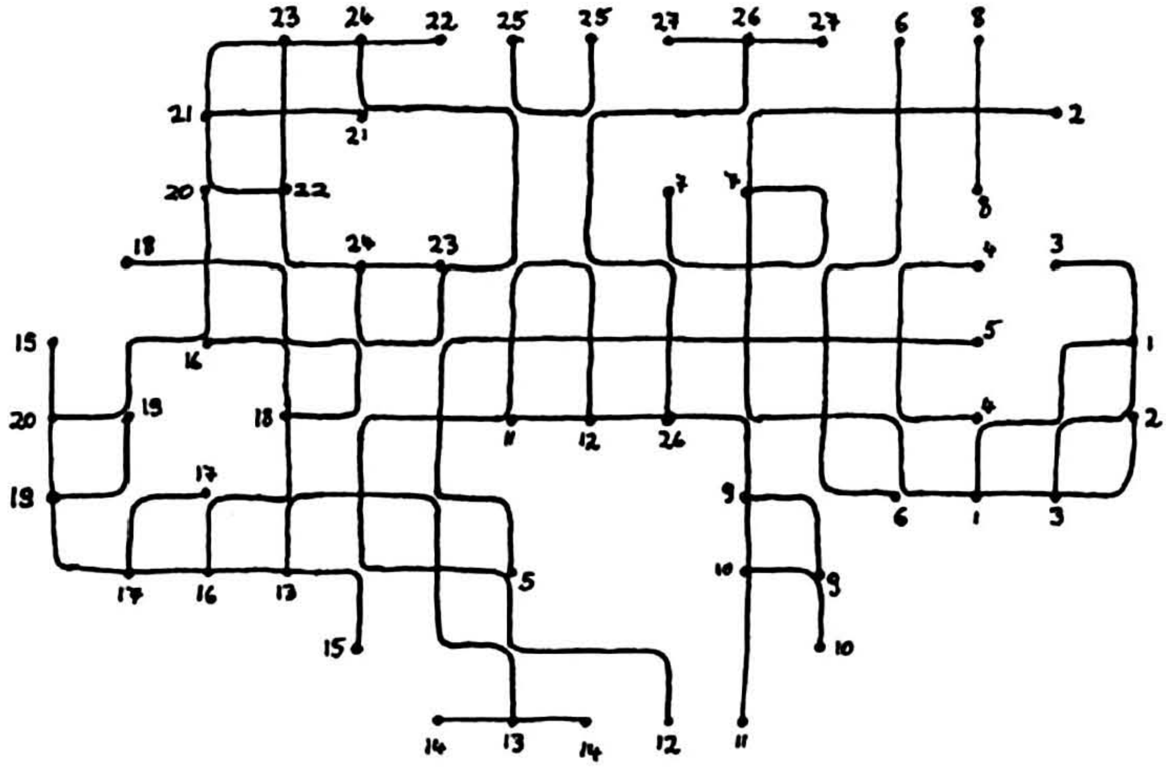


Figure 2b): A solution to the problem of Figure 2a).

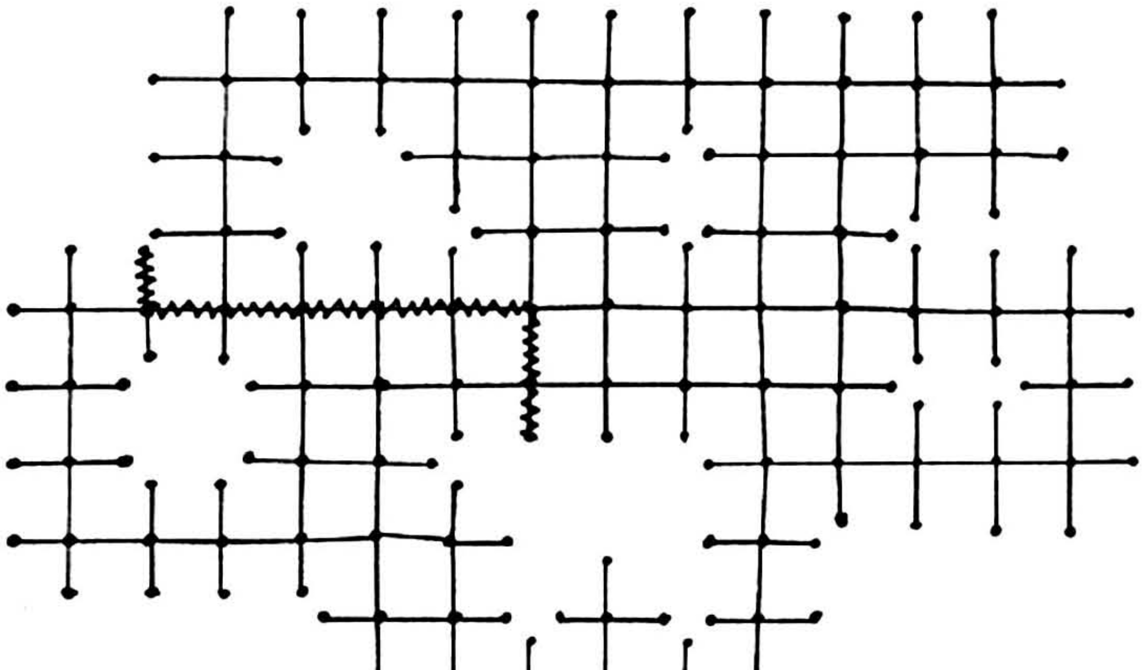
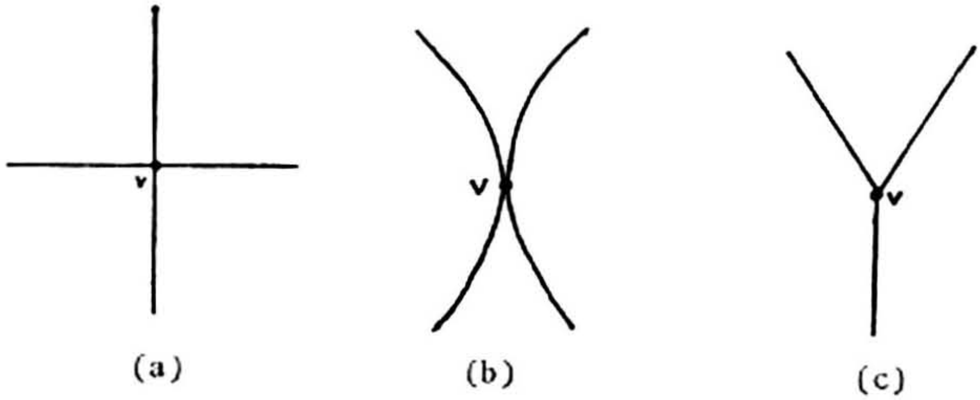
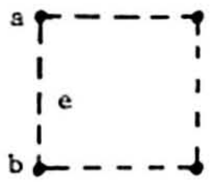


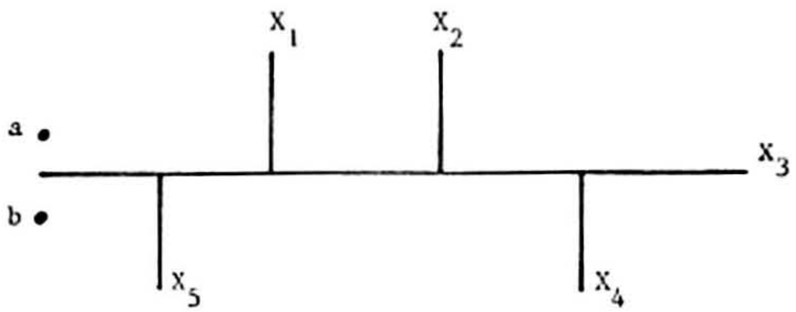
Figure 3: The multiple source dual D(R) of the routing region of Figure 2. A cut of capacity 8 is shown wiggled.



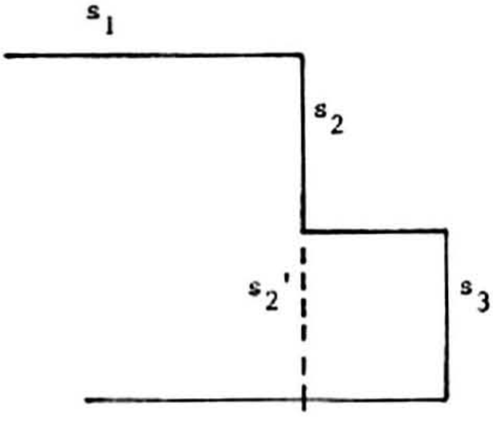
<u>Figure 4:</u> The cuts in a) and b) interfere, the cuts in c) do not interfere.



<u>Figure 5:</u> The four dashed edges are deleted.



<u>Figure 6:</u> X_i is smaller than X_j for $i < j$



<u>Figure 7:</u> The cut $Y = s_1\ s_2\ s_3\ s_4\ \ldots$ and segment s_2'.

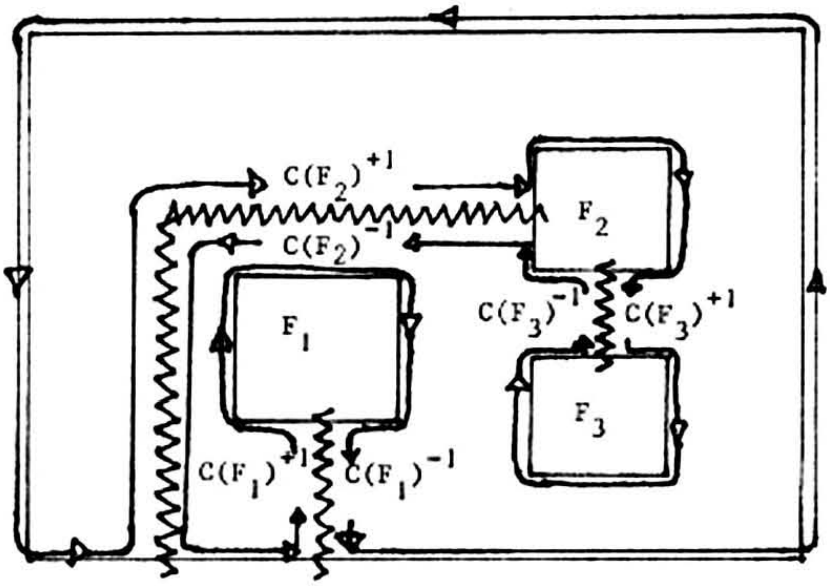


Figure 8: A slicing S of R. We have $\text{parent}(F_3) = F_2$, $\text{parent}(F_1) = \text{parent}(F_o) = F_o$ where F_o is the infinite face. The cyclic ordering of $B \cup C$ is indicated by arrows.

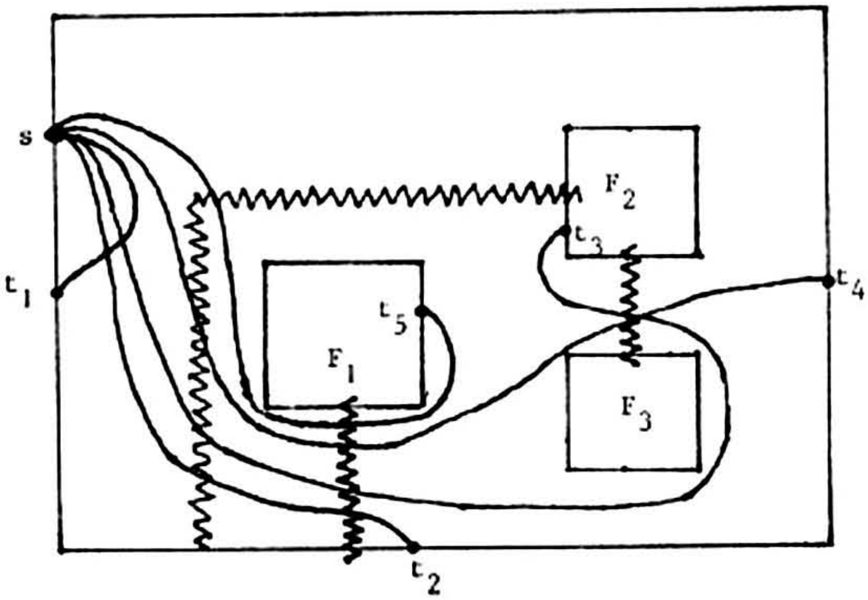


Figure 9: Net N_i connects s and t_i. The elementary pieces of N_5 are $(s, C(F_2)^{+1})$, $(C(F_2)^{-1}, C(F_1)^{+1})$, $(C(F_1)^{-1}, t_5)$ and the elementary pieces of N_4 are $(s, C(F_2)^{+1})$, $(C(F_2)^{-1}, C(F_1)^{+1})$, $(C(F_1)^{-1}, C(F_3)^{-1})$, $(C(F_3)^{+1}, t_4)$. Since $C(F_1)^{-1}$, $C(F_3)^{-1}$, t_5 occur in that order in the cyclic ordering of $B \cup C$ (cf. Figure 8), N_4 is left of N_5. In general, N_i is left of N_j for $i < j$.

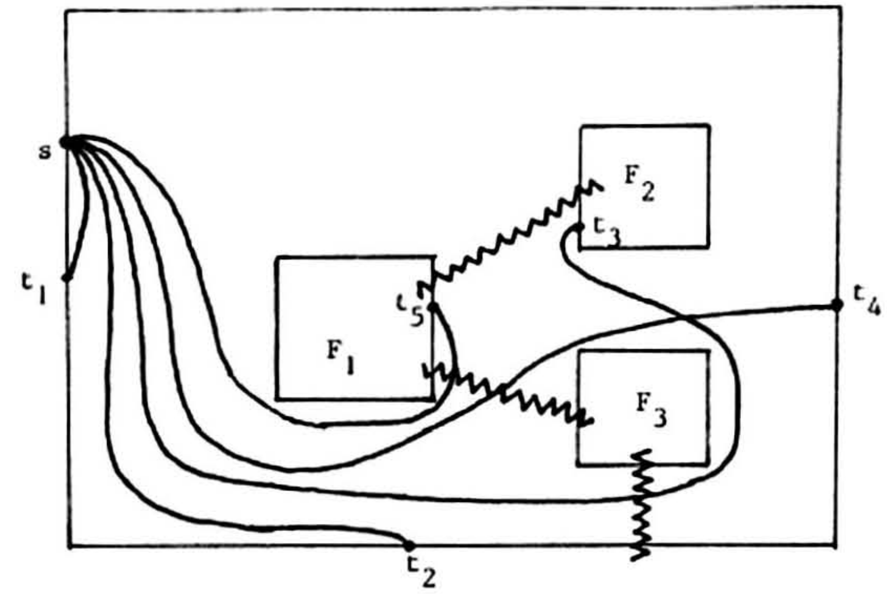


Figure 10: A different slicing of the routing region of Figure 8. The elementary pieces of nets N_5 and N_4 are now: $N_5 = (s, C(F_1)^{+1}), (C(F_1)^{-1}, t_5)$ and $N_4 = (s, C(F_1)^{+1}), (C(F_1)^{-1}, t_4)$. Thus N_4 is left of N_5.

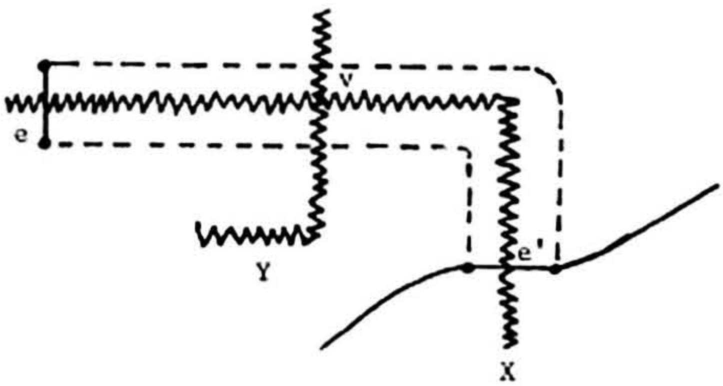


Figure 11: Simple cuts X and Y cross. Net N (to be routed across e) crosses cut X in edge e'.

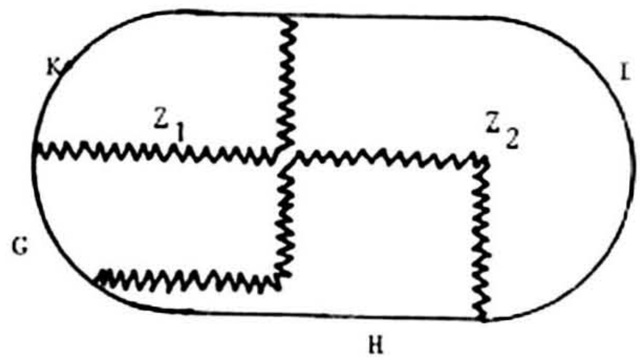


Figure 12: Cuts Z_1 and Z_2 and the partition of $B \cup C$ into four intervals G,H,I,K.

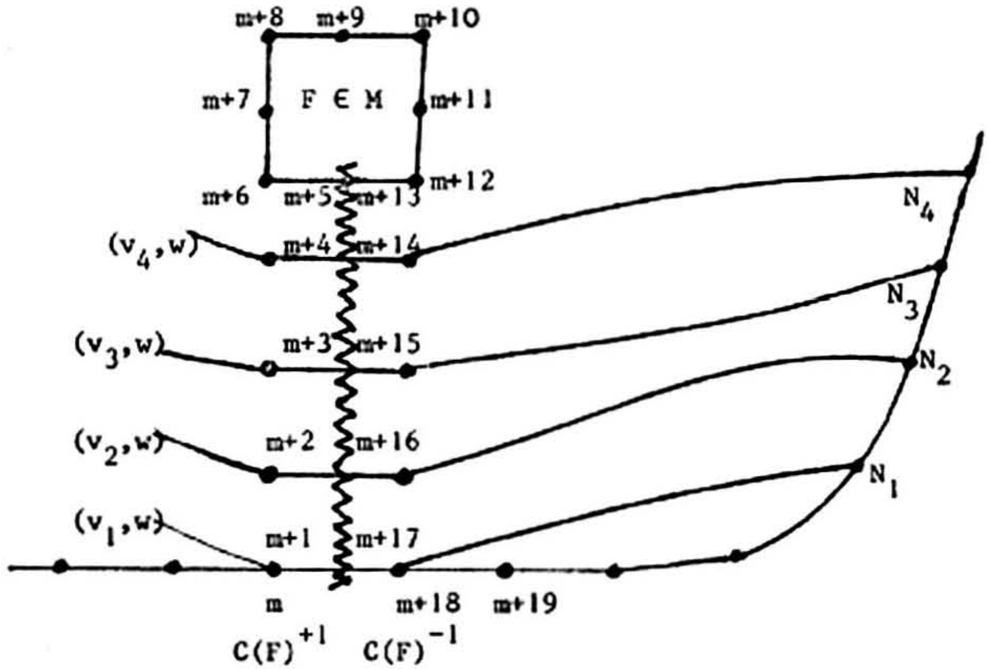


Figure 13: The numbering of the elements in $B \cup C$.
For the nets going across cut $C(F)$ we have chosen local routings which reflect the ordering of the tails of the nets N_1, N_2, N_3 and N_4, i.e. tail (N_i) is left of tail (N_j) for $i < j$. The tail of net N_i is the part of N_i beginning with $C(F)^{-1}$. We represent w by integer m+i in the elementary piece (v_i, w) of net N_i ending in $C(F)^{+1}$.