P-21

Near-Optimal Routing of Multi-Nets Around a Rectangle.

A 1.6 Approximation Algorithm for Routing Multiterminal Nets Around a Rectangle

Teofilo F. Gonzalez
Department of Computer Science
The University of California
Santa Barbara, CA 93106

Sing-Ling Lee
Programs in Computer Science
The University of Texas at Dallas
Richardson, Texas 75080

(title lifted from BioBib)

ABSTRACT

The problem of connecting a set of terminals that lie on the sides of a rectangle to minimize the total area is discussed. We present an $O(n^2)$ approximation algorithm to solve this problem. Our algorithm generates a solution with area ≤ 1.6 * OPT, where OPT is the area of an optimal solution. The nets are routed according to the following greedy strategy: the wire connecting all points from a net is one that crosses the least number of corners. For some nets there are several paths that cross the least number of corners. A subset of these nets is connected by paths that blend with the connecting paths for other nets. The path for the remaining nets is selected using several strategies and 2^6 layouts are obtained. The best of these layouts is the solution generated by our algorithm.

I. INTRODUCTION

Let G be a rectangular component of size h by w (height by width). There are n terminals ($T_1, T_2, \ldots, T_n$) on the sides of G. These terminals are partitioned into m subsets denoted by $N_1, N_2, \ldots, N_m$. Each subset N_i is called a net and it is assumed that $|N_i| > 1$ for all i. The problem depicted in figure 1.1 consists of five nets: $N_1 = \{T_2, T_4, T_7\}$, $N_2 = \{T_1, T_3, T_{10}, T_{14}\}$, $N_3 = \{T_5, T_{12}\}$, $N_4 = \{T_6, T_8, T_9\}$ and $N_5 = \{T_{11}, T_{13}, T_{15}\}$. It is assumed that every pair of terminals is at least $\lambda > 0$ units apart and every terminal is located at least λ units from each of the corners of G. All the terminals in each net must be made electrically common by connecting them with wires. The path followed by these wires can be partitioned into a finite number of straight line segments. Each of these line segments must lie on the same plane as G, be on the outside of G and be parallel to a side of G. Perpendicular line segments can intersect at any point, but parallel line segments must be at least λ units apart. Also, all line segments must be at least λ units away from every side of rectangle G except in the vicinity where a line segment connects a terminal. The *R1M problem* consists of specifying paths for all the interconnections subject to the rules mentioned above in such a way that the total area is minimized, i.e., place the component together with all the wires inside a rectangle (with the same orientation as G) of least possible area. This problem has applications in the layout of integrated circuits ([L] and [R]) and conforms to a set of design rules for VLSI systems [MC].

The 2-R1M is defined similarly, except that all nets are restricted to be of size two. Hashimoto and Stevens present an O(n log n) algorithm to solve the R1M problem for the case when all the points in S lie on one side of G. An Ω(n log n) lower bound on the worst case time complexity for this problem was established in [GLL]. Algorithms to solve the 2-R1M problem appear in [La] and [GL2]. If more than two layers are allowed and wire overlap is permitted, the problem becomes an NP-hard problem [SBR]. Other generalizations of the 2-R1M problem have been shown to be NP-hard [La]. Gonzalez and Lee [GL3] present an $O(n^2)$ approximation algorithm for the R1M problem that generates a solution with area ≤ 1.69 * OPT. In this paper we present an approximation algorithm with a worst case approximation bound of 1.6.

For i=2,3,4, let R1M-i denote a R1M problem in which all global nets contain terminals on exactly i sides of G. In section II we introduce our notation and present some basic results. In order to simplify the exposition of our results, we begin by presenting approximation algorithms for restricted versions of the R1M problem. In section III-V we present an approximation algorithms for the R1M-3, R1M4 and R1M-2 problems. In section VI we combine the results obtained in sections II-V to obtain our

1.6 approximation algorithm for the R1M problem.

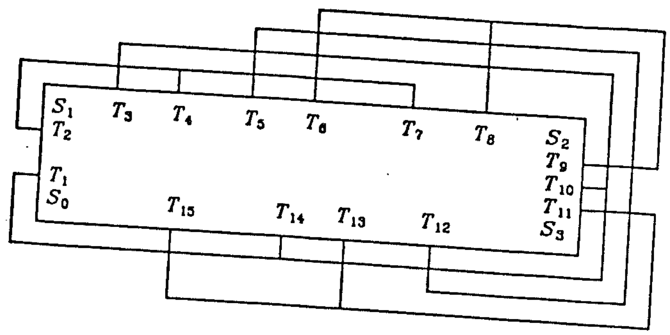


figure 1.1

II. DEFINITIONS AND BASIC RESULTS.

Label the sides of the component (in the obvious way) left, top, right and bottom. Starting in the bottom-left corner of G, traverse the sides of the rectangle clockwise. The ith corner to be visited is labeled S_{i-1}. Assume that the ith terminal visited is terminal T_i . The *close interval* $[x , y]$, where x and y are the corners of G or the terminals T_i, consists of all the points on the sides of G that are visited while traversing the sides of G in the clockwise direction starting at point x and ending at point y. Note that the interval [x , x] consists of only one point. Parentheses are used instead of square brackets for open intervals. We use $[S_0 , S_1]$, $[S_1 , S_2]$, $[S_2 , S_3]$ and $[S_3 , S_0]$; 0, 1, 2 and 3; L, T, R, and B; and L', T', R' and B' to represent the left, top, right and bottom sides of G, respectively. Terminal T_i is said to belong to side l, $S(i) = l$, if T_i is located in $[S_l , S_{(l+1) mod (4)}]$. The function I(j) indicates the index of the net to which terminal T_j belongs. The function L(j) is defined in such a way that the interval $[T_j , T_{L(j)}]$ is the smallest interval that includes all the terminals from net $N_{I(j)}$. Set D $= \{d_1 , d_2 ,..., d_k\}$ is said to be an *assignment* if the cardinality of set D is m and D contains exactly one index of a terminal from each net. Any subset of an assignment is said to be a *partial assignment*. An assignment D indicates the starting point for the path connecting all the points in each net. If i ∈ D then the path connecting all the terminals in net $N_{I(i)}$ starts at terminal T_i moving perpendicular to side S(i) and then it continues in the clockwise direction with respect to G until it reaches $T_{L(i)}$. Each terminal, T_k, in net $N_{I(i)}$ is joined to this path by a line segment perpendicular to side $S(T_k)$. The starting point for some connecting paths might not be defined in a partial assignment. The assignment for the layout given by figure 1.1 is {2 , 3 , 5 , 6 , 11}. For any l ∈ D, we say that the path connecting net $N_{I(l)}$ given by D *crosses* point z if z ∈ $[T_l , T_{L(l)}]$. If i ∈ D, we say that the path connecting net $N_{I(i)}$ *begins* at point T_i and *ends* at point $T_{L(i)}$.

For any assignment (or partial assignment) D we define the *height function* H_D for x,y $\in \{T_1 , T_2 ,..., T_n\} \cup \{S_0 , S_1 , S_2 , S_3\}$ as follows:

$$H_D(x , y) = \max\{ \text{number of paths given by D that cross point } z \mid z \in [x , y] \}.$$

For example, $H_D(S_0 , S_1)$ is 1, $H_D(T_5 , T_5)$ is 3 and $H_D(S_2 , S_3)$ is 3 for the assignment, D, whose layout appears in figure 1.1.

Lemma 2.1: For every assignment D, there is a rectangle Q of size h_Q by w_Q, where

$$h_Q = h + (H_D(S_1 , S_2) + H_D(S_3 , S_0)) * \lambda, \text{ and}$$
$$w_Q = w + (H_D(S_0 , S_1) + H_D(S_2 , S_3)) * \lambda$$

with the property that rectangle G together with the interconnecting paths defined by D can be made to fit inside Q. A layout with area $h_Q * w_Q$ can be constructed in O(n log n) time.

Proof: The proof is a direct generalization of the proof for the 2-R1M problem that appears in [L]. The algorithm that constructs the final layout uses as a subalgorithm the procedure given in [GLL] and [HS].

Net N_i is said to be a ***global net*** if at least two of its terminals are located on opposite sides of G. Net N_i is said to be *local* otherwise, i.e., if all its terminals are located on the same side of G or on two adjacent sides of G. Nets $N_2 = \{ T_1, T_3, T_{10}, T_{14} \}$ and $N_3 = \{ T_5, T_{12} \}$ are the only global nets in the problem depicted in figure 1.1. For assignment D we define the *function A(D)* as

$$(h + (H_D(S_1, S_2) + H_D(S_3, S_0)) * \lambda) * (w + (H_D(S_0, S_1) + H_D(S_2, S_3)) * \lambda)$$

i.e., the total area required for a layout of G and all the interconnections given by D.

Definition 2.1: D'

Let D' be the partial assignment in which all the local nets are connected by paths crossing the least number of corners of G.

Lemma 2.2: Every assignment D can be transformed to an assignment M such that $D' \subseteq$ M and $A(M) \leq A(D)$.

Proof: The proof follows the same lines as the one for the 2-R1M problem that appears in [L].

At this point our algorithm abandons the procedures given in [La] and [GL2]. The main difficulty that we encounter in extending the results for the 2-R1M problem to this problem is that the divide and conquer step seems not applicable. The reason for this is that there seems to be no rule to separate the nets with terminals located in three or four sides of G into groups that can be routed independently of each other.

III. APPROXIMATION ALGORITHM FOR THE R1M-3 PROBLEM.

In this section we present an approximation algorithm for the R1M-3 problem. Let M_3 be the set of global nets with terminals located on exactly three sides of G. Clearly, all global nets belong to set M_3. For the set of nets M_3 we construct assignment D_3 as follows: $D_3 = \{$ all nets in M_3 are connected by paths that cross the least number of corners of G$\}$. Let M_3^{TB} (M_3^{LR}) be the set of all nets in M_3 without terminals located on either the left or right (top or bottom) side of G. Let m_3^{TB} and m_3^{LR} represent the number of elements in M_3^{TB} and M_3^{LR} respectively.

Lemma 3.1: Let D be an optimal assignment such that $D' \subseteq$ D. Let M be D except that all nets in $M_3^{TB} \cup M_3^{LR}$ which are assigned as in our algorithm. Then

$$H_M(S_1,S_2) + H_M(S_3,S_0) \leq H_D(S_1,S_2) + H_D(S_3,S_0) + x_3^{TB}, \text{ and}$$

$$H_M(S_0,S_1) + H_M(S_2,S_3) \leq H_D(S_0,S_1) + H_D(S_2,S_3) + x_3^{LR},$$

where x_3^{TB} (x_3^{LR}) is the number of nets in X_3^{TB} (X_3^{LR}) and X_3^{TB} (X_3^{LR}) contains all the nets in set M_3^{TB} (M_3^{LR}) that are connected differently in D and M.

set	contribution to our lower bound for $h/(\lambda)+H_D(S_1,S_2)+H_D(S_3,S_0)$	contribution to our lower bound for $w/(\lambda)+H_D(S_0,S_1)+H_D(S_2,S_3)$
M_3^{TB}	$1.5m_3^{TB} + 0.5x_3^{TB}$	$2m_3^{TB} + 0.5x_3^{TB}$
M_3^{LR}	$2m_3^{LR} + 0.5x_3^{LR}$	$1.5m_3^{LR} + 0.5x_3^{LR}$

Table 3.1: Lower bounds for the R1M3 problem.

Lemma 3.2: Let D be an optimal assignment such that $D' \subseteq$ D. Then assignment D and rectangle G satisfy the lower bounds given in table 3.1.

Theorem 3.1: For the R1M-3 problem, let D be an optimal assignment such that $D' \subseteq$ D and let M be the assignment generated by our algorithm. Then, $A(M) \leq 1.6 * A(D)$.

IV. APPROXIMATION ALGORITHM FOR THE R1M-4 PROBLEM.

In this section we present an approximation algorithm for the R1M-4 problem. Let M_4 be the set of all global nets with terminals located on the four sides of G and let m_4 represent the number of nets in it. Clearly, all global nets belong to set M_4. Nets N_i and N_j (both in set M_4) are said to be *agreeable* if on some side of G no terminal from net N_i is between any two terminals from net N_j and no terminal from net N_j is between any two terminals from net N_i. Procedure PARTITION defined below partitions the set M_4 into the sets M_4^A, M_4^N, $M_4^{T'}$, $M_4^{R'}$, $M_4^{B'}$, $M_4^{L'}$, M_4^T, M_4^R and M_4^B. Since some of these sets consist of 1-tuples, 2-tuples and 4-tuples, by partition we mean that every element in set M_4 is in one and only one of the tuples in these sets. After the algorithm terminates, let m_4^X represent the number of tuples in M_4^X, for X ∈ { A , N , T' , R' , B' , L' , T , R , B }. It is very important to keep in mind that sets M_4^A , $M_4^{T'}$, $M_4^{R'}$, $M_4^{B'}$ and $M_4^{L'}$ consists of 1-tuples, set M_4^N has only 4-tuples and the remaining sets contain only 2-tuples.

procedure PARTITION

$M_4^A \leftarrow \{ N_i \mid N_i \in M_4$ and $|N_i| = 4 \}$;
W ← M_4 - M_4^A;

Z ← { N_i | N_i ∈ W has at least two terminals located on each side of G};
W ← W - Z;
$M_4^T \leftarrow \phi$; $M_4^R \leftarrow \phi$; $M_4^B \leftarrow \phi$; $M_4^L \leftarrow \phi$; $M_4^N \leftarrow \phi$;

while |Z| ≥ 4 *do*
  let N_i, N_j, N_k and N_l be any four nets in Z;
  if any two of these four nets are agreeable
    then delete from Z two agreeable nets (two out of the four) and add them as a pair to M_4^T if these two nets are agreeable on the left or right side of G, otherwise add these two nets as a pair to M_4^R
    else delete all four nets from Z
      $M_4^N \leftarrow M_4^N \cup \{ (N_i, N_j, N_k, N_l) \}$
  endif;
endwhile;

// later on we explain what to do when Z ≠ φ //

while there is a net in Z with at least two terminals located on side X and exactly one terminal located on the remaining sides of G *do*
  Let y be one of such nets and X be the side on which it has more than one terminal;
    Z ← Z - { y };
    $M_4^X \leftarrow M_4^X \cup \{ y \}$;
endwhile;

W ← W ∪ Z;
Z ← { N_i | N_i ∈ W has at least two terminals located on the top side if G};
W ← W - Z;

while there are two nets in Z with exactly one terminal located on the bottom side of G *do*
  delete two of such nets from Z and add them as a pair to M_4^T;
endwhile;

while there are two nets in Z with exactly one terminal located on the same side of G *do*
  delete two of such nets from Z and add them as a pair to M_4^T
endwhile;

// later on we explain what to do when Z ≠ φ //

Z ← { N_i | N_i ∈ W has at least two terminals located on the right side of G };
W ← W - Z;

while there are two nets in Z with exactly one terminal located on the left side of G *do*
  delete two of such nets from Z and add them as a pair to M_4^R;
endwhile;

while |Z| ≥ 2 *do*
  delete any two nets from Z and add them as a pair to M_4^R
endwhile;

// later on we explain what to do when Z ≠ φ //

Z ← { N_i | N_i ∈ W has at least two terminals located on the bottom side of G };
W ← W - Z;

while |Z| ≥ 2 *do*
  remove any two nets from Z and add them to M_4^B as a pair;
endwhile;

// later on we explain what to do when $Z \neq \phi$ //

end of procedure

For some problem instances the algorithm fails to assign some nets to any of the sets M_4^X, $X \in \{ A, N, T', R', B', L', L, T, R, B \}$. Let us assume that this is not the case, i.e., Z is always empty whenever we reach each of the "// later on ... //" comments in our procedure. In section VI we explain what to do when this is not the case.

a) Construction of the Assignment D_A''.

In what follows we explain how the assignment D_A'' for all the nets in set M_4^A is to be constructed. Figure 4.1 gives a layout for the assignment D_A'' constructed by our procedure for a set M_4^A with four nets. For any permutation, π, of the nets in set M_4^A and an integer i ($0 \leq i \leq 3$) we define an assignment (denoted ASG(π,i)) as follows: the first net in π is connected by a path that begins on side i and ends on side (i+3)mod(4); and the path connecting the kth net in π begins on side j and ends on side (j+3)mod(4), where j is the side on which the path connecting the (k-1)st net in π ends. We claim that for every integer i ($0 \leq i \leq 3$) there is a permutation, π (that depends on i) of the nets in set M_4^A such that there is a layout for assignment ASG(π,i) with the property that for any k ($1 < k \leq m_4^A$) the path connecting the (k-1)st net in π and the path connecting the kth net in π can share the same track on the side where the path connecting the (k-1)st net ends. In this case we say that π *is a valid permutation with respect to i*. We claim that every integer i, $0 \leq i \leq 3$, there is a valid permutation for the set of nets in M_4^A. Assignment D_A'' is just ASG(π , 1).

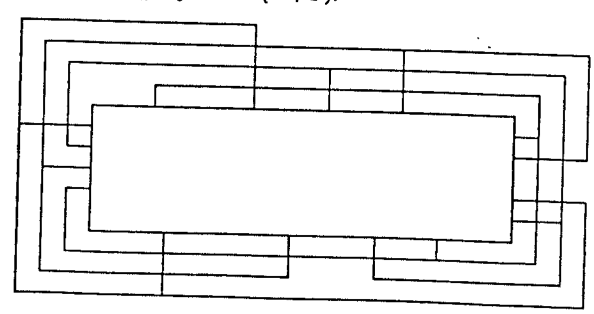


Figure 4.1: Assignment D_A'' for the case when $m_4^A = 4$.

b) Construction of the Assignment D_N''.

The assignment D_N'' is constructed by applying the following rule to each 4-tuple in M_4^N. Let (N_i, N_j, N_k , N_l) be any tuple in set M_4^N. Let us assume that these nets have been ordered in such a way that N_i is the net (amongst these four nets) whose bottommost terminal located on the left side of G is closest to the top side of G; net N_j is the net (amongst N_j, N_k and N_l) whose rightmost terminal located on the bottom side of G is closest to the left side of G; and net N_l is the net (amongst N_k and N_l) whose leftmost terminal located on the top side of G is closest to the right side of G. A layout for the assignment constructed for these nets is given in figure 4.2.

c) Construction of assignment D_X'' for $X \in \{ T', R', B', L' \}$.

Assignment D_X'', $X \in \{ T', R', B', L' \}$, is constructed by a procedure similar to the one used in part (a). For brevity we will not discuss it.

d) Construction of assignment D_X'' for $X \in \{ T, R, B \}$.

Assignment D_X'', $X \in \{ T, R, B \}$, for the set M_4^X is constructed by applying the following rule to each 2-tuple in M_4^X. Let (N_i , N_j) be any 2-tuple in set M_4^X. From algorithm PARTITION we know that both of these nets are agreeable on at least one side of G because both of these nets have exactly one pin located on some side of G. These nets are assigned in such a way that there is a layout in which these nets share the same track on side Z of G and both nets have exactly one terminal located on side Z. Side Z is selected using the following priorities: the opposite side of X has the highest priority; and the adjacent sides of G have the lowest priorities. Note that side Z cannot be the same as side X.

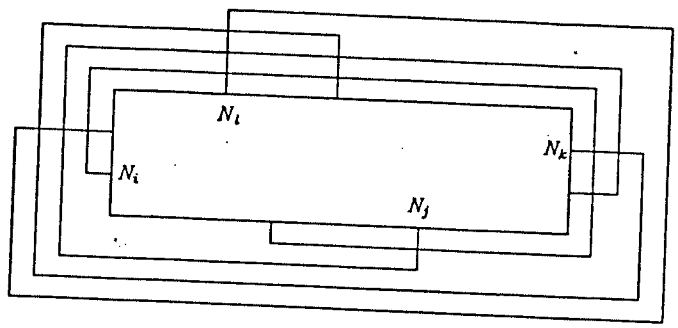


Figure 4.2: Assignment for a tuple in M_4^N.

e) Final Assignment and Proof of Approximation Bound.

Let $D_4 = D_A'' \cup D_N'' \cup D_{T'}'' \cup D_{R'}'' \cup D_{B'}'' \cup D_{L'}'' \cup D_T'' \cup D_R'' \cup D_B''$. In what follows we assume that m_4^A is a multiple of four. In section VI we indicate what to do when this is not the case.

Lemma 4.1: Let D be an optimal assignment such that $D' \subseteq$ D. Let M be D except that all the nets in $M_4^A \cup M_4^N$ are routed as in our approximation algorithm. Then

$H_M(S_1,S_2) + H_M(S_3,S_0) \leq H_D(S_1,S_2) + H_D(S_3,S_0) + 0.5m_4^A + 1 + 7x * M_4^N$, and

$H_M(S_2,S_3) + H_M(S_0,S_1) \leq H_D(S_2,S_3) + H_D(S_0,S_1) + 0.5m_4^A + 7y * M_4^N$, where x and y are nonnegative reals such that x + y = 1.

In what follows we assume that m_4^X, X ∈ { T' , R' , B' , L' }, is a multiple of four. In section VI we indicate what to do when this is not the case.

Lemma 4.2: Let D be an optimal assignment such that $D' \subseteq$ D. Let M be D except for all the nets in M_4^X, X ∈ { T' , R' , B' , L' }, are routed as in our approximation algorithm. Then

$$H_M(S_1,S_2) + H_M(S_3,S_0) \leq H_D(S_1,S_2) + H_D(S_3,S_0) + a * m_4^X, \text{ and}$$

$$H_M(S_2,S_3) + H_M(S_0,S_1) \leq H_D(S_2,S_3) + H_D(S_0,S_1) + b * m_4^X, \text{ where}$$

$$a = \begin{cases} 3 & \text{if } X \in \{T', B'\} \\ 2 & \textit{otherwise}; \end{cases} \quad \text{and} \quad b = \begin{cases} 2 & \text{if } X \in \{T', B'\} \\ 3 & \textit{otherwise}. \end{cases}$$

For X ∈ { T , R , B } let p_4^X be the fraction of pairs in M_4^X that are connected in D_X'' by paths that do not overlap on the top or bottom side of G, and $q_4^X = 1 - p_4^X$. Since all pairs of nets in M_4^B are agreeable on the top side of G, we know that $q_4^B = 0$.

Lemma 4.3: Let D be an optimal assignment such that $D' \subseteq$ D. Let M be D except that all nets in M_4^X, X ∈ { T, R, B }, are routed as in our approximation algorithm. Then

$$H_M(S_1,S_2) + H_M(S_3,S_0) \leq H_D(S_1,S_2) + H_D(S_3,S_0) + p_4^X * m_4^X + 2q_4^X * m_4^X,$$

$$H_M(S_2,S_3) + H_M(S_0,S_1) \leq H_D(S_2,S_3) + H_D(S_0,S_1) + 2p_4^X * m_4^X + q_4^X * m_4^X.$$

set	contribution to our lower bound for $h/(\lambda)+H_D(S_1,S_2)+H_D(S_3,S_0)$	contribution to our lower bound for $w/(\lambda)+H_D(S_0,S_1)+H_D(S_2,S_3)$
M_4^A	$2.5m_4^A$	$2.5m_4^A$
M_4^N	$14m_4^N$	$14m_4^N$
$M_4^{T'} \cup M_4^{B'}$	$4m_4^{T'} + 4m_4^{B'}$	$4.5m_4^{T'} + 4.5m_4^{B'}$
$M_4^{R'} \cup M_4^{L'}$	$4.5m_4^{R'} + 4.5m_4^{L'}$	$4m_4^{R'} + 4m_4^{L'}$
$M_4^T \cup M_4^B$	$p_4^T * m_4^T + 5m_4^T + 6m_4^B$	$q_4^T * m_4^T + 6m_4^T + 6m_4^B$
M_4^R	$p_4^R * m_4^R + 6m_4^R$	$q_4^R * m_4^R + 5m_4^R$

Table 4.1: Lower bounds for the R1M-4 problem.

Lemma 4.4: Let D be an optimal assignment such that $D' \subseteq$ D. Then assignment D and rectangle G satisfy the lower bounds given in table 4.1.

Theorem 4.1: For the R1M-4 problem, let D be an optimal assignment such that $D' \subset D$ and let M be the assignment generated by our algorithm. Then, $A(M) \le 1.6 * A(D)$.

V. APPROXIMATION ALGORITHM FOR THE R1M-2 PROBLEM.

In this section we present an approximation algorithm for the R1M-2 problem. Clearly, all global nets contain terminals located on only two opposite sides of G. Let M_2^{TB} (M_2^{LR}) represent the set of global nets with terminals located only on the top and bottom (left and right) sides of G. Let m_2^{TB} and m_2^{LR} represent the number of nets in M_2^{TB} and M_2^{LR} respectively. Assume that the number of elements in each of these sets is a multiple of 6. In section VI we indicate the modifications that need to be made to the algorithm when this is not the case. First, let us explain (informally) how the nets in these sets are routed. We will explain only the procedure for M_2^{TB}, since the one for M_2^{LR} is similar. The set M_2^{TB} is partitioned into six equally sized groups. Later on we explain precisely how this partition is obtained. A group is said to be routed by a path type L (R) if all the nets in this group are routed by a path that does not cross the right (left) side of G. We will construct 2^6 assignments. Each of these assignments corresponds to a string of six elements from the alphabet { L , R }. The ith element in a string specifies the type of path used to connect all nets in the ith group. Let us now formally define this construction process. For each net N_j in M_2^{TB} (M_2^{LR}), let p(j) be the index of the leftmost (bottommost) terminal of N_j located on the top (left) side of G. Let $P^{TB} = \{ p(j) \mid N_j \in M_2^{TB} \}$, and $P^{LR} = \{ p(j) \mid N_j \in M_2^{LR} \}$. Each of these sets is partitioned into the sets N_i^{TB} and N_i^{LR} for $0 \le i \le 5$ as follows:

N_i^{TB} = { I(j) | j is the kth smallest value in the set P^{TB} and $(i/5)*|P^{TB}| < k \le ((i+1)/5)*|P^{TB}|$ } and

N_i^{LR} = { I(j) | j is the kth smallest value in the set P^{LR} and $(i/5)*|P^{LR}| < k \le ((i+1)/5)*|P^{LR}|$ }.

We define the following sets for $0 \le i \le 63$:

D_i^{TB} = { if in the binary representation of i the $(l+1)$st least significant bit is one then the connecting path for each net in set N_l^{TB} does not cross corner S_2, otherwise the connecting path for such a net does not cross corner S_1 | $0 \le l \le 5$ }, and

D_i^{LR} = { if in the binary representation of i the $(l+1)$st least significant bit is one then the connecting path for each net in set N_l^{LR} does not cross corner S_1, otherwise the connecting path for such a net does not cross corner S_0 | $0 \le l \le 5$ }.

For $0 \le i,j \le 63$, define assignment $D_{i,j}$ as $D_{i,j} = D' \cup D_i^{TB} \cup D_j^{LR}$, and let P be one of the $D_{i,j}$'s of least area.

In lemma 5.1 we assume that when interchanging the connecting paths of two nets in M_2^{TB} that cross on the top side of G will increase by at most 2 the vertical height of the assignment. This assumption is not always true. In figure 5.1 we show one counterexample to this assertion. We call this interchange a *type I interchange*. The two nets involved in this interchange form a *type I pair*. We also assume that when interchanging the connecting path for a net in M_2^{TB} to one that does not cross the left or right sides of G, will increase the vertical height of an assignment by at most two. This statement always holds true.

Lemma 5.1: Let D be an optimal assignment such that $D' \subseteq D$. There is an assignment $D_{i,j}$ (constructed by our algorithm) such that if M is defined as D except for all the nets in $M_2^{TB} \cup M_2^{LR}$ which are routed as in the assignment $D_{i,j}$, then

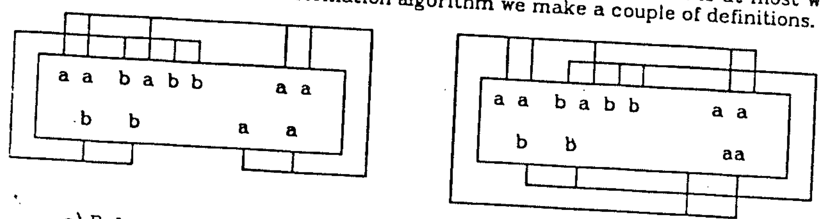
a) $H_M(S_1,S_2) + H_M(S_3,S_0) \le H_D(S_1,S_2) + H_D(S_3,S_0) + (0.6) * m_2^{TB}$

b) $H_M(S_0,S_1) + H_M(S_2,S_3) \le H_D(S_0,S_1) + H_D(S_2,S_3) + (0.6) * m_2^{LR}$.

It is assumed that when D is transformed to any of the assignments $D_{i,j}$ and two paths that cross on the top side of G are interchanged, such an interchange is not a type I interchange.

Let L be an optimal solution that includes D'. The existence of at least one of these assignments is guaranteed by lemma 2.2. Let S be any of the $D_{i,j}$ assignments. The difference between S and L is the way in which some nets in $M_2^{TB} \cup M_2^{LR}$ are routed. Lemma 5.1 shows that at least one of our assignments differs in vertical height from L by at most $1.6 * m_2^{TB}$ and in horizontal height by at most $1.6 * m_2^{LR}$. As noted in the text appearing immediately before lemma 5.1, we cannot yet claim that the 1.6 bound holds when type I interchanges occur.

In order for our 1.6 bound to hold true we will transform each assignment S (each of the $D_{i,j}$'s) to another assignment R such that if the total increase in vertical height when transforming L to S by interchanging the nets in M_2^{LR} was $\leq w + 3\alpha$, where α is the number of type I interchanges made and w is the contribution from all other interchanges, then the actual difference in vertical height between L and R is at most $w + 2\alpha$. Before presenting our transformation algorithm we make a couple of definitions.



a) Before the interchange

b) After the interchange

Fig. 5.1: Type I interchange

Definition 5.1: A pair of nets, (a,b), in M_2^{TB} is said to form a *crossing pair* in an assignment if

i) Net "a" is connected by a path that does not cross the right side of G.
ii) Net "b" is connected by a path that does not cross the left side of G.
iii) On the bottom side of G all the terminals from net "b" appear to the left of all the terminals form net "a". And
iv) On the top side of G there is at least one terminal from net "a" located to the right of the rightmost terminal from net "b" and at least one terminal from net "a" is located to the left of the leftmost terminal from net "b".

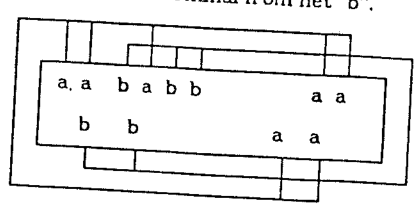


Fig. 5.2: Crossing pair (a,b).

One can easily prove that any two nets involved in a type I interchange will form a crossing pair. Later on we show that such a pair has harming effects only when it is a conflicting pair.

Definition 5.2: A crossing pair (a,b) *includes* point x if either of the two following conditions is satisfied:

i) If x is located on the top side of G, then all terminals from net "b" located on the top side of G appear to the left of x and the rightmost terminal from net "a" is located to the right of x. Or
ii) If x is located on the bottom side of G, then all terminals from net "b" located on the bottom side of G appear to the left of x and all the terminals from net "a" are located to the right of x.

Definition 5.3: A crossing pair (a,b) *partially includes* point x if x is located on the bottom side of G and either of the two following conditions is satisfied:
i) All terminals from net "a" located on the bottom side of G are located to the right of x and the leftmost terminal from net "b" located on the bottom side of G is not located to the right of point x. Or
ii) All terminals from net "b" located on the bottom side of G are located to the left of x and the rightmost terminal from net "a" located on the bottom side of G is not located to the left of point x.

Note that if point x is included in a crossing pair then it is also partially included in it, but the converse is not always true.

Definition 5.4: A *conflicting pair*, (a,b), is a crossing pair that includes the leftmost point with maximum height located on the top side of G, partially includes all the points with maximum height located on the bottom side of G and includes either the leftmost or the rightmost point with maximum height located on the bottom side of G.

Our postprocessing procedure will find conflicting pairs and interchange their connecting paths. In figure 5.3 we show a conflicting pair and in figure 5.4 we indicate how these paths are interchanged. This transformation reverses the effects of type I interchanges.

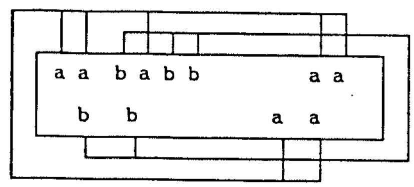


Fig. 5.3: (a,b) forms a conflicting pair.

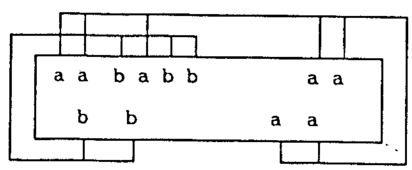


Fig. 5.4: Interchange of the conflicting pair given in figure 5.3.

algorithm MODIFY
  R ← S;
  while there is a conflicting pair in R *do*
    interchange a conflicting pair in R
  endwhile
end of algorithm

Let α be the number of type I interchanges made when transforming S to L and let w + 3α be the total increase in vertical height because of such transformation.

Lemma 5.2: The vertical height of assignment R minus the vertical height of assignment L is at most w + 2α.

set	contribution to our lower bound for $h/(\lambda)+H_D(S_1,S_2)+H_D(S_3,S_0)$	contribution to our lower bound for $w/(\lambda)+H_D(S_0,S_1)+H_D(S_2,S_3)$
M_2^{TB}	m_2^{TB}	$2m_2^{TB}$
M_2^{LR}	$2m_2^{LR}$	m_2^{LR}

Table 5.1: Lower bounds for the R1M-2 problem.

An algorithm similar to MODIFY has to be applied to the $D_{i,j}$ in order to be able to claim the bound 1.6 * M_2^{LR}. Let us refer to this new procedure MODIFY-LR. Our algorithm works as follows:

Obtain assignments $D_{i,j}$;
Use MODIFY and MODIFY-LR to transform each $D_{i,j}$ into a new assignment, $M_{i,j}$;
Output the assignment $M_{i,j}$ of least area;

Lemma 5.3: Let D be an optimal assignment such that $D' \subset$ D. There is an assignment $M_{i,j}$ (constructed by our algorithm) such that if M is defined as D except for all the nets in $M_2^{TB} \cup M_2^{LR}$ which are routed as in the assignment $M_{i,j}$, then

a) $H_M(S_1,S_2) + H_M(S_3,S_0) \le H_D(S_1,S_2) + H_D(S_3,S_0) + (0.6) * m_2^{TB}$

b) $H_M(S_0,S_1) + H_M(S_2,S_3) \le H_D(S_0,S_1) + H_D(S_2,S_3) + (0.6) * m_2^{LR}$.

Lemma 5.4: Let D be an optimal assignment such that $D' \subseteq$ D. Then assignment D and rectangle G satisfy the lower bounds given in table 5.1.

Theorem 5.1: Let D be an optimal assignment such that $D' \subseteq D$ and let P be the assignment produced by our algorithm. Then, $A(P) \leq 1.6 * A(D)$.

VI. APPROXIMATION ALGORITHM FOR THE R1M PROBLEM.

In this section we show that our algorithm takes $O(n^2)$ time and generates a solution with area $\leq 1.6 *$ OPT, where OPT is the area of an optimal solution.

algorithm for the R1M Problem

Construct assignments D', D_3 and D_4;
$D_{i,j} \leftarrow D' \cup D_3 \cup D_4 \cup D_i^{TB} \cup D_j^{LR}$ for $0 \leq i,j \leq 63$;
Apply algorithm MODIFY and MODIFY-LR to each $D_{i,j}$ to obtain $M_{i,j}$;
Let P be one of the $M_{i,j}$'s of least area;
Construct and output a layout with area A(P) for P;

end of algorithm

Theorem 6.1: The time complexity of our procedure is $O(n^2)$.

Theorem 6.2: Let D be an optimal assignment such that $D' \subseteq D$ and let P be the assignment produced by our algorithm. Then, $A(P) \leq 1.6 * A(D)$.

Our algorithm generates 2^{12} assignments and it outputs one that requires the least layout area. An algorithm that only generates 2^6 assignments can be easily obtained by only taking the best of the modified $D_3 \cup D_4 \cup D_i^{TB}$ together with the best of the modified $D_3 \cup D_4 \cup D_i^{LR}$. For brevity we will not prove that this solution also satisfies our approximation bounds. In sections IV-V we assumed that the number of nets in some sets was a multiple of some fixed constant. More specifically, we might have to delete at most 5 nets from sets M_2^{TB} and M_2^{LR}; and at most 23 nets from M_4. All of these nets with a "small" number of terminals can be routed optimally by trying all possible routing paths and then selecting the best of the solutions generated. If some of these nets do not have a "small" number of terminals then select any routing paths for them, note that it will not make too much difference which routing path is selected since their contribution to the lower bound in an optimal solution is large (contribution from the number of terminals). There are better ways of dealing with the remaining 33 nets, however for brevity these other methods will not be discussed in this paper.

VII. REFERENCES

[AHU] Aho, A. , Hopcroft, J. and Ullman, J. , " *The Design and Analysis of Computer Algorithms* ", Addison-Wesley, 1975.

[GL1] Gonzalez, T. and Lee S., "An Optimal Algorithm for Optimal Routing Around a Rectangle," Proceeding of the 20th Annual Allerton Conference on Comm, Control and Computing, U. of Illinois, Oct. 1982, 636-645.

[GL2] Gonzalez, T. and Lee S., "An O(n log n) Algorithm for Optimal Routing Around a Rectangle," Technical Report #116, Programs in Computer Science, The University of Texas at Dallas, Nov 1982.

[GL3] Gonzalez, T. and Lee S., "Routing Multiterminal Nets Around a Rectangle," Technical Report #202, Programs in Computer Science, The University of Texas at Dallas, Nov 1983.

[GLL] Gupta, U. I., Lee D. T. and Leung J., "An Optimal Solution for the Channel-Assignment Problem", *IEEE Transactions on Computers*, Vol. C-28, No. 11, 1979, pp. 807-810.

[HS] Hashimoto, A. and Stevens J. E., "Wire Routing by Optimizing Channel Assignment without large Apertures", Proceedings of the 8th IEEE Design Automation Conference, 1971, pp 155-169.

[L] LaPaugh, A. S., "A Polynomial Time Algorithm for Optimal Routing Around a Rectangle", Proceedings of the 21st IEEE Foundations of Computer Science, 1980, pp 282-293.

[La] LaPaugh, A. S., "Algorithms for Integrated Circuit Layout, An Analytic Approach", Ph.D. dissertation, Massachusetts Institute of Technology, 1980.

[MC] Mead, C. and Conway, L., " *Introduction to VLSI Systems*", Addison- Wesley, 1980.

[R] Rivest, R.L., "The PI (Placement and Interconnect) System", Proceedings of the 19th IEEE Design Automation Conference, pp 475-481.

[SBR] Sahni, S., Bhatt, A. and Raghavan, R., "The Complexity of Design Automation Problems", Technical Report 80-23, Department of Computer Science, The University of Minnesota, Dec 1980.