Routing Multiterminal Nets Around a Rectangle

TEOFILO F. GONZALEZ AND SING-LING LEE

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 $O(nm)$ approximation algorithm to solve this problem where n is the number of terminals and m is the number of signal nets. Our algorithm generates a solution with an area $\leq 1.69^*$ OPT where OPT is the area of an optimal solution. Our algorithm routes some of the nets by a simple greedy strategy. The remaining nets are routed using several strategies and four layouts are obtained. The best of these layouts is the solution generated by our algorithm.**

Index Terms — **Algorithms, greedy methods, minimize area, VLSI, wire routing.**

I. Introduction

LET T be a rectangle and S be a set of points that lie on the sides of T. Let $N_1, N_2, \cdots, N_m$ be any partition of set S. Each subset N_i is called a net. All the points in each net have to be made electrically common by interconnecting them with wires. The path followed by these wires consists of a finite number of horizontal and vertical line segments. These line segments can be assigned to two different layers. All the horizontal line segments are assigned to one layer and all the vertical ones are assigned to the other layer. Line segments on different layers can be connected at any given point z by a wire perpendicular to the layers if both line segments cross point z on their respective layers. Every pair of distinct and parallel line segments must be at least λ units apart and every line segment must be at least λ units from each side of T, except in the region where the path joins the point in S it connects. Also, no path is allowed inside T on any of the layers.

Problem R1M (routing around one module) consists of specifying paths for all the connecting wires in such a way that the total area is minimized. That is, to place T together with all the connecting wires (that must satisfy the restrictions imposed above) inside a rectangle (with the same orientation as T) of least possible area. This problem has applications in the layout of integrated circuits [3] and [10] and conforms to a set of design rules for VLSI systems [9].

The case when each net is restricted to be of size two has been studied in great detail. We call this problem the 2-R1M problem. In [2] and [5] an $O(n \log n)$ algorithm is presented to solve the R1M problem for the case when all the points in S lie on one side of T where n is the number of elements in set S. An $\Omega(n \log n)$ lower bound on the worst case time complexity for the 2-R1M problem was established in [5]. Algorithms to solve the 2-R1M problem appear in [4] and [6]. The one in [6] is optimal with respect to the time complexity bound. When many layers are available and wire overlap (for wires in different layers) is permitted, the problem becomes NP-hard [11]. Other generalizations of the 2-R1M problem have also been shown to be NP-hard [4]. In this paper we present an $O(nm)$ approximation algorithm to solve the R1M problem. Our algorithm generates a solution with area $\leq 1.69 * \text{OPT}$, where OPT is the area of an optimal solution. An approximation algorithm that generates solutions within 60 percent of optimal appears in [8]. The difference between these two algorithms is that the one presented in this paper is easier to understand and takes less time to execute. Also, the proof for the approximation bound of 1.6 is much more complex.

The first few steps of our procedure follow the initial steps of the algorithms in [4] and [6]. These steps consist of reducing our problem to the problem of selecting the starting point (by default the direction is clockwise) for the connecting path of each net and specifying the starting point for the connecting paths of all local nets (a local net is one in which all the terminals appear on the same side of T or on two adjacent sides of T). The algorithm routes some of the nets by a simple greedy strategy. The remaining nets are routed using several strategies producing a nonempty set of feasible solutions. One of the layouts with least area will be the solution generated by our algorithm.

In Section II we present our basic results and the algorithm. The analysis of our algorithm will be presented in Section III.

II. Basic Results and the Algorithm

We begin by defining the R1M problem and introducing some useful notation similar to the one in [6]. Then we present some basic results and our algorithm. The analysis of our algorithm will be presented in Section III.

Let T be a rectangular component of size h by w (height by width). There are n terminals $(T_1, T_2, \cdots, T_n)$ on the sides of T. These terminals are partitioned into m subsets denoted by $N_1, N_2, \cdots, N_m$. Each subset N_i is called a net. The problem depicted in Fig. 1 consists of the following 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 away from each of the corners of T. 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

Manuscript received May 22, 1984; revised May 9, 1985. This work was supported in part by the National Science Foundation under Grant DCR-8503163.

T. F. Gonzalez is with the Department of Computer Science, University of California, Santa Barbara, CA 93106.

S. -L. Lee is with the Department of Computer Science, Pennsylvania State University, University Park, PA 16802.

IEEE Log Number 8608654.

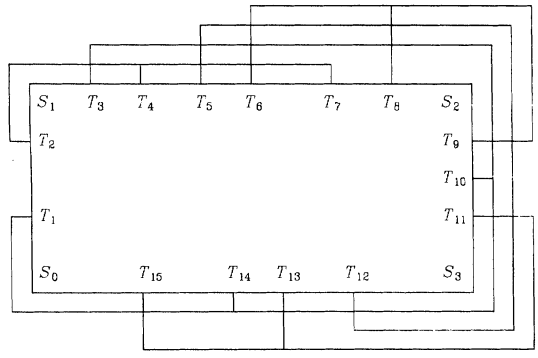


Fig. 1.

number of straight line segments. Each of these line segments must lie on the same plane as T, be on the outside of T and be parallel to a side of T. 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 T 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 interconnecting wires inside a rectangle (with the same orientation as T) of least possible area.

Label the sides of the component (in the obvious way) left, top, right and bottom. Starting in the bottom-left corner of T, 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 T or the terminals T_i, consists of all the points on the sides of T that are visited while traversing the sides of T in the clockwise direction starting at point x and ending at point y. Parentheses are used instead of square brackets for open ended intervals. We use $[S_0, S_1]$, $[S_1, S_2]$, $[S_2, S_3]$ and $[S_3, S_0]$ to represent the left, top, right, and bottom sides of T, respectively. Terminal T_i is said to belong to side l, $S(i) = l$, if T_i is located in $[S_l, S_{(l+1) \bmod(4)}]$.

The function $I(j)$ indicates the index of the net to which terminal T_j belongs. The function $L(j)$ is such that the interval $[T_j, T_{L(j)}]$ is the smallest interval that includes all the terminals from net $N_{I(j)}$. Set $D = \{d_1, d_2, \cdots, d_n\}$ is said to be an *assignment of directions* if D contains exactly one index from a terminal in each net. Any subset of an assignment is said to be a *partial assignment*. An assignment D indicates the starting point for the connecting path of all nets (the direction is by default clockwise). If $i \in D$ then the connecting path of all the terminals in net $N_{I(i)}$ starts at terminal T_i moving perpendicular to side $S(i)$ and then it moves in the clockwise direction with respect to T until it reaches $T_{L(i)}$. Each of the remaining terminals, T_k, in net $N_{I(i)}$ are joined to this path by a line segment perpendicular to side $S(T_k)$ of T. In a partial assignment, the starting point of some connecting paths might not be specified. The assignment for the layout given by Fig. 1 is $\{2, 3, 5, 6, 11\}$. For any $l \in D$, we say that the connecting path for $N_{I(l)}$ given by D *crosses* point z if $z \in [T_l, T_{L(l)}]$.

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

$H_D(x, y)$ = max{number of paths given by D that cross point $z \mid z \in [x, y]$}. Let D be the assignment for the layout given in Fig. 1. For assignment D we have that $H_D(S_0, S_1)$ is 1, $H_D(T_5, T_5)$ is 3, and $H_D(S_2, S_3)$ is 3. We shall refer to $H_D(x, y)$ as the *height of assignment D on the interval* $[x, y]$ in T.

The next two lemmas establish that the R1M problem reduces to the problem of finding an assignment D with least

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

and then in $O(n \log n)$ time ($O(n)$ time if the set of terminals is initially sorted) one can construct a final layout for it.

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

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

and

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

with the property that rectangle T together with the interconnecting paths defined by D can be made to fit inside Q.

Proof: The proof is a direct generalization of the proof for the 2-R1M problem that appears in [4]. ■

Lemma 2.2: A final layout with the area given by Lemma 2.1 can be obtained in $O(n \log n)$ time for any assignment D.

Proof: The proof of this lemma is a straightforward generalization of the proof for the 2-R1M problem that appears in [4]. The algorithm that constructs the final layout uses a subalgorithm the procedure given in [5] and [2]. ■

Net N_i is said to be a *global net* if at least two terminals in N_i are located on opposite sides of T. Net N_i is said to be *local*, i.e., if all its terminals are located on the same side of T or on adjacent sides of T. The problem shown in Fig. 1 has N_2 and N_3 as the only global nets. 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 by the layout of T together with all the interconnections specified 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 T.

Lemma 2.3: There is an optimal assignment D such that $D' \subseteq D$.

Proof: The proof follows the same lines as the one for the 2-R1M problem that appears in [4]. ■

Lemma 2.3 shows that for any instance of the R1M problem there exists an optimal solution in which all local nets are connected by paths crossing at most one corner of T. The R1M problem has been reduced to the problem of finding the direction and the starting point for the connecting paths of all global nets in the presence of the partial assignment D'. At this point our procedure differs from the one given in [4] and [6]. 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 possible. The reason for this is that there seems to be no rule to partition the nets with terminals located in three or four sides of T.

It should be clear that it is only required to specify the paths connecting the global nets since local nets are connected as indicated in D'. Also, once we have an assignment one can use the proof of Lemma 2.2 (a constructive proof) to find the final layout. In what follows we explain how to obtain our final assignment.

Let m_4 be the number of global nets with terminals on the four sides of T and let M_4 be the set of all these nets. Assume that m_4 is a multiple of 2. In Section IV we indicate how to modify our algorithm when m_4 is not a multiple of 2. Nets N_i and N_j (both in set M_4) are said to be *agreeable* if on some side of T no terminal in N_i is between any two terminals in N_j and no terminal in N_j is between any two terminals in N_i. The following procedure defines the sets M_4^C and M_4^N, a partition of set M_4. After the procedure terminates, let m_4^C and m_4^N denote the number of pairs (of nets) in M_4^C and M_4^N, respectively.

procedure Construct
  $W \leftarrow M_4$; $M_4^N \leftarrow \phi$; $M_4^C \leftarrow \phi$;
  while there are two nets in W with exactly one terminal located on the same side of T **do**
    Let N_i and N_j be such nets;
    $M_4^N \leftarrow M_4^N \cup \{(N_i, N_j)\}$;
    $W \leftarrow W - \{N_i, N_j\}$;
  endwhile
  while $W \neq \phi$ **do**
    Let N_i and N_j be any two nets in W;
    $M_4^C \leftarrow M_4^C \cup \{(N_i, N_j)\}$;
    $W \leftarrow W - \{N_i, N_j\}$;
  endwhile
end of procedure

It is simple to see that the two nets in each tuple in M_4^N are agreeable. The two nets in each tuple in M_4^N are routed as in Fig. 2(a) (this figure shows the case when the nets have exactly one terminal located on the top side of T). Every pair of nets in M_4^C is routed as in Fig. 2(b). We should point out that there is at most one net in set M_4^C that contains only one terminal on the top side of T. This is also true for all of the other sides of T. It is important to remember this fact when we establish a lower bound for the area of an optimal solution (Lemma 3.4).

Let D'' = {all nets with terminals located on exactly three sides of T are connected by the path that crosses two corners of T} $\cup$ {nets N_i and N_j such that $(N_i, N_j) \in M_4^N$ are routed in such a way that they share the same track on a side in which the two nets have exactly one terminal (Fig. 2(a) gives the routing for two nets with exactly one terminal located on the top side of T)} $\cup$ {nets N_i and N_j such that $(N_i, N_j) \in M_4^C$ are routed as shown in Fig. 2(b)}.

The global nets that have not yet been assigned in $D' \cup D''$ are the ones with terminals located on only two opposite sides of T. 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 T. 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 3. In Section IV we indicate the modifications that need to be made to the algorithm if this is not the case. For each 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 T. Let $P^{TB} = \{p(j) \mid N_j \in N_{M_2^{TB}}\}$ 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 $0 \leq i \leq 2$ as follows:

$$N_i^{TB} = \{I(j) \mid j \text{ is the } k\text{th smallest value in the set } P^{TB} \text{ and } (i/3)|P^{TB}| < k \leq ((i+1)/3)|P^{TB}|\}$$

and

$$N_i^{LR} = \{I(j) \mid j \text{ is the } k\text{th smallest value in the set } P^{LR} \text{ and } (i/3)|P^{LR}| < k \leq ((i+1)/3)|P^{LR}|\}.$$

We define the following sets for $0 \leq i \leq 7$:

D_i^{TB} = {if in the binary representation for i the $(l+1)$st least significant bit is one then the connecting path for each net in set N_i^{TB} does not cross corner S_2; otherwise, the connecting path for such a net does not cross corner $S_1 \mid 0 \leq l \leq 2$}

and

D_i^{LR} = {if in the binary representation for i the $(l+1)$st least significant bit is one then the connecting path for each net in set N_i^{LR} does not cross corner S_1; otherwise, the connecting path for such a net does not cross corner $S_0 \mid 0 \leq l \leq 2$}.

At this point we construct assignment $D_{i,j}$ for $0 \leq i, j \leq 7$ as follows:

$$D_{i,j} = D' \cup D'' \cup D_i^{TB} \cup D_j^{LR}.$$

The algorithm will output the layout of one of the assignments with least area. In the next section we show that the area of the layout generated by our algorithm is $\leq 1.69 *$ OPT, where OPT is the area of an optimal assignment.

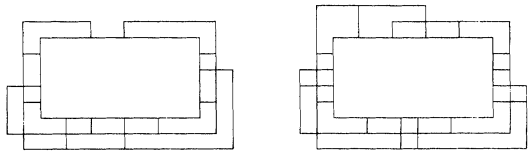


Fig. 2.

III. Analysis of the Algorithm

In this section we show that our algorithm generates a solution with area $\leq 1.69 *$ OPT where OPT is the area of an optimal solution and we show that the time complexity of our algorithm is $O(nm)$. Before showing that our algorithm generates a solution with objective function value within 69 percent of the optimal solution value, we need to introduce some useful notation and prove some initial results.

Let M_3^{TB} (M_3^{LR}) be the set of all nets with terminals on only three sides of T and without terminals located on either the

left or right (top or bottom) sides of T. Let m_3^{TB} and m_3^{LR} represent the number of nets in M_3^{TB} and M_3^{LR}, respectively. Our approximation algorithm routes these nets by a path that crosses the least number of corners of T. In Fig. 3(a) we show the connecting path for one net in set M_3^{LR}. Suppose that this net is routed as shown in Fig. 3(b) in an optimal assignment D. Let M be D except for the connecting path for the net that appears in Fig. 3(b) is connected as shown in Fig. 3(a). It is simple to show that

$$H_M(S_1, S_2) + H_M(S_3, S_0) \leq H_D(S_1, S_2) + H_D(S_3, S_0)$$

and

$$H_M(S_0, S_1) + H_M(S_2, S_3) \leq H_D(S_0, S_1) + H_D(S_2, S_3) + 1.$$

A straightforward generalization of the above observation is given by Lemma 3.1.

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

$$H_M(S_1, S_2) + H_M(S_3, S_0) \leq H_D(S_1, S_2) + H_D(S_3, S_0) + m_3^{TB}$$

and

$$H_M(S_0, S_1) + H_M(S_2, S_3) \leq H_D(S_0, S_1) + H_D(S_2, S_3) + m_3^{LR}.$$

Proof: We only prove the bound for M_3^{TB} since the proof for M_3^{LR} is similar. The contribution from a net in M_3^{LR} to $(H_D(S_1, S_2) + H_D(S_3, S_0))$ and $(H_D(S_0, S_1) + H_D(S_2, S_3))$ is at least one if such a net is routed differently than in our algorithm. When we interchange such a net so that it is routed as in our algorithm, the increase in the horizontal height $(H_D(S_0, S_1) + H_D(S_2, S_3))$ is at most 1 and there is no increase in the vertical height $(H_D(S_1, S_2) + H_D(S_3, S_0))$. Hence, the sum of all such contributions is given by the bounds given in the statement of the lemma. This completes the proof of the lemma. ■

Lemma 3.2: Let D be an optimal assignment such that $D' \subseteq D$. Let M be D except for the set of nets in $M_4^N \cup M_4^C$ are routed as in our approximation algorithm. Then

$$H_M(S_1, S_2) + H_M(S_3, S_0) \leq H_D(S_1, S_2) + H_D(S_3, S_0) + (1 + x) * m_4^N + 2m_4^C$$

and

$$H_M(S_0, S_1) + H_M(S_2, S_3) \leq H_D(S_0, S_1) + H_D(S_2, S_3) + (1 + y) * m_4^N + 2m_4^C$$

where x, y are nonnegative real values such that $x + y = 1$.

Proof: It is simple to show that the proof for this lemma follows from the following claim.

Claim: Let D be an optimal assignment such that $D' \subseteq D$.

a) Let M be assignment D except for the nets N_i and N_j, such that $(N_i, N_j) \in M_4^N$, are routed as in our approximation algorithm. Then

$$H_M(S_1, S_2) + H_M(S_3, S_0) \leq H_D(S_1, S_2) + H_D(S_3, S_0) + 1 + a$$

and

$$H_M(S_0, S_1) + H_M(S_2, S_3) \leq H_D(S_0, S_1) + H_D(S_2, S_3) + 1 + (1 - a)$$

where a is 0 or 1.

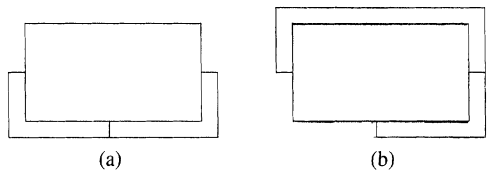


Fig. 3.

b) Let M be assignment D except for the nets N_i and N_j, such that $(N_i, N_j) \in M_4^C$, are routed as in our approximation algorithm. Then

$$H_M(S_1, S_2) + H_M(S_3, S_0) \leq H_D(S_1, S_2) + H_D(S_3, S_0) + 2$$

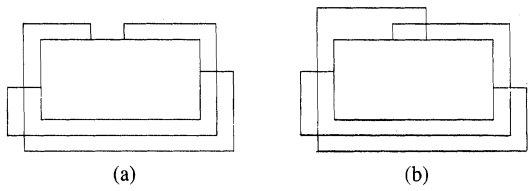
and

$$H_M(S_0, S_1) + H_M(S_2, S_3) \leq H_D(S_0, S_1) + H_D(S_2, S_3) + 2.$$

Proof of Claim: It is simple to see that any net with terminals on the four sides of T will contribute in assignment D at least one unit to the total horizontal height $(H_D(S_0, S_1) + H_D(S_2, S_3))$ and will also contribute at least one unit to the total vertical height $(H_D(S_1, S_2) + H_D(S_3 + S_0))$. The contribution to the vertical height and the horizontal height for a pair of nets in M_4^N routed by our algorithm is either 3 and 4, or 4 and 3 (the former case is when the two nets are routed by sharing a track on the top or bottom sides of T and the latter one occurs when the sharing is on the left or right side of T). Therefore, when the path connecting two nets in M_4^N is interchanged to the paths given in assignment M, the maximum amount we increase the vertical height and the horizontal height is given by a). This completes the proof for Part a). Using similar arguments one can easily prove Part b). This concludes the proof of the claim and the lemma. ■

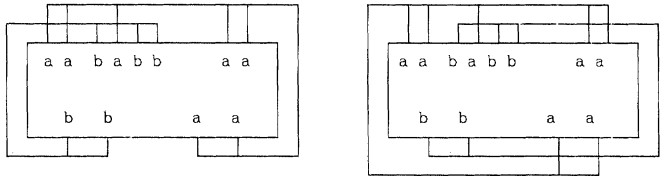
For simplicity, let us assume that in all optimal assignments the nets in $M_2^{TB} \cup M_2^{LR}$ are routed by paths that cross exactly two corners of T. In general, this is not true, however, our results can also apply to these problem instances by doing a simple modification to our analysis. When transforming an optimal assignment D to one of the assignments generated by our algorithm, we will first modify assignment D to one in which all nets in $M_2^{TB} \cup M_2^{LR}$ are connected by paths that cross exactly two of the corners of T. This new assignment is obtained by connecting all nets in M_2^{TB} (M_2^{LR}) that are connected by paths that cross the four corners of T, by paths that do not cross the right (top) side of T. The new assignment satisfies the restrictions imposed before. Note that each time an interchange is performed we will increase by one the height of one side of T, but we will decrease the height of another side (adjacent to the previous one) of T by one. In the final analysis we must account for the above changes in D. For simplicity, this more detailed analysis is omitted. Readers interested in a more detailed analysis are referred to [8].

In Lemma 3.2 we assume that when interchanging the paths connecting two nets in M_2^{TB} that cross on the top side of T will increase by at most 2 the vertical height of the assignment. This assumption is not always true. In Fig. 5 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*. Interchanging the connecting



(a) (b)

Fig. 4.



(a) (b)

Fig. 5. Type I interchange. (a) Before the interchange. (b) After the interchange.

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. Sometimes the result given by Lemma 3.3 will hold true even when type I interchanges occur when transforming an optimal assignment to the ones generated in our algorithm. Since there are cases when this will not be true we need to apply some postprocessing improvement to eliminate the effects of the type I interchanges. For brevity we will not explain the postprocessing procedure in this paper. An interested reader can find it in [7]. It is important to note that such procedure takes $O(nm)$ time. We should point out that after applying our postprocessing procedure the 1.69 approximation bound will hold true for all problem instances.

Lemma 3.3: 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 assignments $D_{i,j}$, then

a) $H_M(S_1, S_2) + H_M(S_3, S_0) \leq H_D(S_1, S_2) + H_D(S_3, S_0) + (2/3) * m_2^{TB}$

and

b) $H_M(S_0, S_1) + H_M(S_2, S_3) \leq H_D(S_0, S_1) + H_D(S_2, S_3) + (2/3) * m_2^{LR}$.

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 T are interchanged, such an interchange is not a type I interchange.

Proof: Since the proof for Part b) is similar to the proof of Part a), we only prove Part a). Before proving our result we make the following observations:

1) The interchange of one connecting path for a net in M_2^{TB} in assignment D to one that crosses the least number of corners in T will increase by at most two the value of $H_D(S_1, S_2) + H_D(S_3, S_0)$.

2) The interchange of two connecting paths for nets in M_2^{TB} that cross on the top side of T in assignment D will increase by at most two the value of $H_D(S_1, S_2) + H_D(S_3, S_0)$.

Let G_i, for $1 \leq i \leq 3$, be the partition of the nets generated by our algorithm (these sets were called N_i^{TB}, for $0 \leq i \leq 2$, in our algorithm). For assignment D, let l_i be the number of nets in G_i that are connected by a path that crosses the left side of T and let r_i be the number of nets in G_i that are not connected by a path that crosses the left side of T. In what follows we assume that there are no type I interchanges.

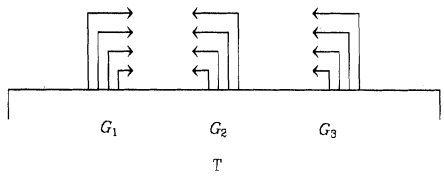


Fig. 6.

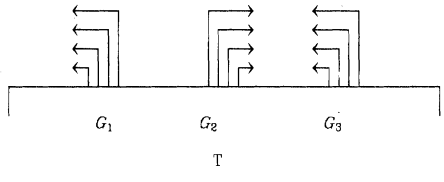


Fig. 7.

If we transform D by interchanging the paths given in Fig. 6 we have that the total increase of the vertical height is given by

$$2 * \max\{r_1, l_2 + l_3\}.$$

If this value is $\leq (2/3) * m_2^{TB}$ then the lemma holds true. Let us assume that it is not the case. Now, it cannot be that $r_1 \geq l_2 + l_3$ because it would imply that $r_1 > (1/3) * m_2^{TB}$, a contradiction. Therefore, the lemma does not hold true when

$$l_2 + l_3 > (1/3) * m_2^{TB}$$

which is equivalent to

$$l_3 > r_2. \tag{1}$$

If we transform D by interchanging the paths given in Fig. 7 we have that the increase of the vertical height is given by the following formula.

$$2 * l_1 + 2 * \max\{r_2, l_3\}.$$

If this value is $\leq (2/3) * m_2^{TB}$ then the lemma holds true. Let us assume that it is not the case. Substituting (1) in the above inequalities we know that these two solutions do not satisfy the lemma when

$$l_1 + l_3 > (1/3) * m_2^{TB}. \tag{2}$$

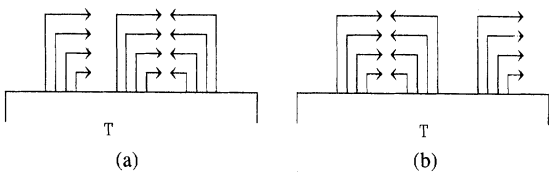
Using similar arguments and the interchanges depicted in Fig. 8(a) and (b), we know that the lemma does not hold true when

$$r_1 + r_3 > (1/3) * m_2^{TB}.$$

Combining this inequality with inequality (2) we have that the number of terminals in G_1 and G_3 is greater than $(2/3) * m_2^{TB}$. This is a contradiction. Hence, at least one of the above solutions satisfies the lemma. This completes the proof of the lemma. ■

Before proving our main result we establish a lower bound on the area required by any optimal solution. This is given by the following lemma.

Lemma 3.4: Let D be an optimal assignment such that $D' \subseteq D$. Then



(a) (b)

Fig. 8.

i) $h/(\lambda) + H_D(S_1, S_2) + H_D(S_3, S_0) \geq m_2^{TB} + 2m_2^{LR} + 1.5m_3^{TB} + 2m_3^{LR} + 5m_4^N + 7m_4^C$.

ii) $w/(\lambda) + H_D(S_0, S_1) + H_D(S_2, S_3) \geq m_2^{LR} + 2m_2^{TB} + 1.5m_3^{LR} + 2m_3^{TB} + 5m_4^N + 7m_4^C$.

Proof: We only prove Part i) since the proof of Part ii) is similar. It is simple to prove the following lower bounds:

1) Each net in $M_2^{LR} \cup M_3^{LR}$ has at least one terminal on the left and right sides of T.

2) Each net in M_3^{TB} has at least one terminal on the left or right side of T.

3) Each pair of nets in M_4^N have at least two terminals on the left and right sides of T.

4) Each pair of nets in M_4^C have at least four terminals on the left and right sides of T, except for at most two of these nets which have at least three terminals on the left and right sides of T.

Using the above observations together with the fact that every terminal is at least λ units away from each corner of T, we get the following bound.

$$h/(\lambda) \geq m_2^{LR} + 0.5m_3^{TB} + m_3^{LR} + 4m_4^C + 2m_4^N.$$

Let us now establish a lower bound on the number of paths crossing the corners of T. Since every path for a net in $M_2^{TB} \cup M_2^{LR} \cup M_3^{TB} \cup M_3^{LR}$ crosses at least two corners of T and the paths connecting every pair of nets in $M_4^N \cup M_4^C$ cross at least six (each path crosses three corners) corners of T, we have that

$$\sum_{i=0}^{3} H_D(S_i, S_i) \geq 2 * (m_2^{LR} + m_2^{TB} + m_3^{LR} + m_3^{TB}) + 6 * (m_4^N + m_4^C).$$

It is simple to show that

$$H_D(S_1, S_2) + H_D(S_3, S_0) \geq (H_D(S_1, S_1) + H_D(S_2, S_2))/2 + (H_D(S_3, S_3) + H_D(S_0, S_0))/2.$$

The lower bound i) follows from the above inequalities. This completes the proof of the lemma. ■

Theorem 3.1: Let D be an optimal assignment such that $D' \subseteq D$ and let P be the solution generated by our algorithm. Then, $A(P) \leq 1.69 * A(D)$.

Proof: Let M be any of the assignments constructed by the algorithm such that $M = D_{i,j}$ and $D_{i,j}$ is an assignment that satisfies Lemma 3.3. We now prove that $A(M) \leq 1.69 * A(D)$. Since $A(P) \leq A(M)$ the result holds true. Using the Lemmas 3.1 to 3.4 we have the following inequalities:

$$A(M)/A(D) \leq (1 + ((2/3) * m_2^{TB} + m_3^{TB} + (1 + x) * m_4^N + 2m_4^C)/(m_2^{TB} + 2m_2^{LR} + 1.5m_3^{TB} + 2m_3^{LR} + 5m_4^N + 7m_4^C)) *$$
$$(1 + ((2/3) * m_2^{LR} + m_3^{LR} + (1 + y) * m_4^N + 2m_4^C)/(m_2^{LR} + 2m_2^{TB} + 1.5m_3^{LR} + 2m_3^{TB} + 5m_4^N + 7m_4^C))$$

where x and y are nonnegative reals and $x + y = 1$.

After multiplying the expression we obtain the expression $(ax + by + \cdots)/(a'x + b'y + \cdots)$ where $a, b, \cdots, a', b', \cdots$ are constants and $x, y, \cdots$ are pairs of m's. At this point it is only required to prove that $a/a', b/b', \cdots$ are all ≤ 1.69. Note that if any of x or y or $\cdots$ are zero, then the corresponding term can be deleted and if all of them are zero then our algorithm generates an optimal solution. The remaining part of the proof is omitted since it involves tedious algebraic manipulations. ■

Theorem 3.2: The time complexity of the algorithm presented in the previous section is $O(nm)$.

Proof: Procedure Construct is the only part of our algorithm (with the exception of the postprocessing procedure) for which an $O(n)$ upper bound on the time complexity is not obvious. In this procedure, the only step that violates this bound is the test performed in the first while loop. An $O(n)$ time bound can be obtained by showing that at each iteration the test can be performed in constant time. Every time we will ignore (perhaps by deleting temporarily from W) all nets in W with at least two terminals located on each side of T. The test is performed by taking any five of the remaining nets. Note that at least two of these five nets will have one terminal on the same side of T. When there are less than five nets, the test can be done by trying all possibilities. Therefore, the total time required by the first while loop is $O(n)$. The postprocessing procedure suggested just before Lemma 3.3 can be carried out in $O(nm)$ time [8]. This completes the proof of the theorem. ■

IV. Conclusions

We have shown that there is an efficient approximation algorithm that generates a solution within 69 percent of optimal for the R1M problem. The algorithm takes $O(nm)$ time and the constant associated with this bound is small. Our algorithm generates 64 assignments and it outputs one that requires the least layout area. We should point out that it is trivial to modify our algorithm so that only 4 assignments (more than 4 partial assignments will be generated) are generated. We omitted this for brevity. A 1.6 approximation algorithm for our problem has been obtained in [8]. The 1.6 algorithm is much more complex and takes longer to execute. Also, the proof for the 1.6 approximation bound is much more elaborate. In Section II we assumed that the number of nets in some sets was a multiple of some fixed constant. If this is not the case, we will route all nets with a "small" number of terminals optimally by trying all possible routing paths and then selecting the best of the solutions generated. For nets without a "small" number of terminals we will select any routing path since their contribution to our lower bound for an optimal solution is large (contribution from the number of terminals). There are better ways of dealing with the remaining 3 nets. However, for brevity the other methods will not be discussed.

REFERENCES

[1] A. Aho, J. Hopcroft, and J. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA: Addison-Wesley, 1975.

[2] A. Hashimoto and J. E. Stevens, "Wire routing by optimizing channel assignment without large apertures," in *Proc. 8th IEEE Design Automation Conf.*, 1971, pp. 155–169.

[3] A. S. LaPaugh, "A polynomial time algorithm for optimal routing around a rectangle," in *Proc. 21st IEEE Foundations Comput. Sci.*, 1980, pp. 282–293.

[4] ——, "Algorithms for integrated circuit layout, an analytic approach," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, 1980.

[5] U. I. Gupta, D. T. Lee, and J. Leung, "An optimal solution for the channel-assignment problem," *IEEE Trans. Comput.*, vol. C-28, pp. 807–810, Nov. 1979.

[6] T. Gonzalez and S. Lee, "An optimal algorithm for optimal routing around a rectangle," in *Proc. 20th Ann. Allerton Conf. Comm. Contr. Comput.*, Urbana, IL, Oct. 1982, pp. 636–645.

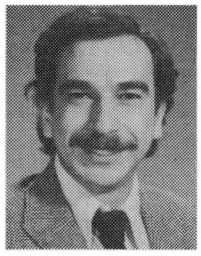
[7] ——, "An $O(n \log n)$ algorithm for optimal routing around a rectangle," Dep. Comput. Sci., Univ. Calif., Santa Barbara, Tech. Rep. TRC 85-12, July 1985.

[8] ——, "A 1.6 approximation algorithm for routing multiterminal nets," Programs Comput. Sci., Univ. Texas, Dallas, Tech. Rep. 204, May 1984, also in *Proc. 22nd Ann. Allerton Conf.*, pp. 498–507.

[9] C. Mead and L. Conway, *Introduction to VLSI Systems*. Reading, MA: Addison-Wesley, 1980.

[10] R. L. Rivest, "The PI (placement and interconnect) system," in *Proc. 19th IEEE Design Automation Conf.*, pp. 475–481.

[11] S. Sahni, A. Bhatt, and R. Raghavan, "The complexity of design automation problems," Dep. Comput. Sci., Univ. Minnesota, Minneapolis, Tech. Rep. 80-23, Dec. 1980.



Teofilo F. Gonzalez was born in Monterrey, Mexico, on January 26, 1948. He received the B.S. degree in computer science from the Instituto Tecnologico de Monterrey, in 1972 and the Ph.D. degree in computer science from the University of Minnesota, Minneapolis, in 1975.

He has been a member of the Computer Science faculty at the University of Oklahoma, the Pennsylvania State University, the Institut Tecnologico de Monterrey, and the University of Texas at Dallas. He is now Associate Professor of Computer Science at the University of California, Santa Barbara. His major research interests are in the design and analysis of algorithms, algorithms and complexity issues in computer-aided design, and scheduling theory.

Dr. Gonzalez is a member of the Association for Computing Machinery and the Operations Research Society of America.

Sing-Ling Lee received the B.A. degree in mathematics from the National Central University, Chung-Li, Taiwan, in 1976, and the M.Sc. and Ph.D. degrees in mathematics from the University of Texas, Dallas, in 1981 and 1985, respectively.

He is currently an Assistant Professor of Computer Science at the Pennsylvania State University. His research interests include the design and analysis of algorithms and VLSI layout.

Dr. Lee is a member of the Association for Computing Machinery.