May 1986

UILU-ENG-86-2211
ACT-69

COORDINATED SCIENCE LABORATORY
College of Engineering
Applied Computation Theory

A NEW METHODOLOGY FOR VLSI LAYOUT

M. Sarrafzadeh
F. P. Preparata

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release, distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-86-2211 (ACT69)	5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab., University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION Semiconductor Research Corporation
6c. ADDRESS (City, State and ZIP Code) 1101 W. Springfield Avenue Urbana, IL 61801	7b. ADDRESS (City, State and ZIP Code) P.O. Box 12053 Research Triangle Park, NC 27709
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Semiconductor Research Corporation 8b. OFFICE SYMBOL (If applicable) N/A	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER SRC-RSCH 84-06-049-6
8c. ADDRESS (City, State and ZIP Code) P.O. Box 12053 Research Triangle Park, NC 27709	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO.: N/A PROJECT NO.: N/A TASK NO.: N/A WORK UNIT NO.: N/A
11. TITLE (Include Security Classification) A New Methodology for VLSI Layout	
12. PERSONAL AUTHOR(S) Sarrafzadeh, Majid and Preparata, Franco	
13a. TYPE OF REPORT Interim Technical, final 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) May 1986 15. PAGE COUNT 20
16. SUPPLEMENTARY NOTATION N/A	
17. COSATI CODES FIELD / GROUP / SUB. GR.	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) VLSI layout, modules, routing

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

We propose a novel technique for solving the VLSI layout problem. The strategy is to recursively interconnect a set of modules, in conformity with the design rules. The basic step consists of merging a pair of strongly-connected modules. An optimal algorithm for routing around two modules, and variations thereof, are presented; the running time is O(nlogn) where n is the total number of terminals. The whole layout is provably wirable with three conducting layers.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code) 22c. OFFICE SYMBOL

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

SECURITY CLASSIFICATION OF THIS PAGE

A NEW METHODOLOGY FOR VLSI LAYOUT *

by

M. Sarrafzadeh and F. P. Preparata

Coordinated Science Laboratory and

Department of Electrical and Computer Engineering

University of Illinois at Urbana-Champaign

Abstract : We propose a novel technique for solving the VLSI layout problem. The strategy is to recursively interconnect a set of modules, in conformity with the design rules. The basic step consists of merging a pair of strongly-connected modules. An optimal algorithm for routing around two modules, and variations thereof, are presented; the running time is O(nlogn) where n is the total number of terminals. The whole layout is provably wirable with three conducting layers.

* This work was supported in part by the Semiconductor Research Corporation under contract RSCH 84-06-049-6.

1. Introduction

Despite the remarkable recent progress in VLSI research, much is still to be done to fully take advantage of the resources in the VLSI environment. The efficiency of new techniques must be evaluated against the inherent limitations of VLSI.

A (digital) circuit is viewed as a *computation map*, that is, a set of (concrete) *modules* which have unalterable geometric shapes (each being itself an interconnection of active devices), and a set of *nets*, each of which specifies a subset of points, called *terminals*, on the boundaries of the modules. Circuit layout (or simply, layout) consists of an embedding of the computation map in a grid graph and of defining the precise conductor paths (external to the modules) necessary to interconnect the terminals as specified by the nets (grid layout). The second step of the circuit layout process, called *wiring*, is the conversion of the grid layout to an actual configuration of wires in a three-dimensional grid, with a bounded (small) number of planes in the third dimension.

The measure of the quality of a given solution to a layout problem is the efficiency with which the corresponding computation map can be laid out in conformity with specific design rules. Prevalent criteria of optimality are minimum area (or space) and minimum wire length (implying minimum signal delay).

Following a common approach, we partition the problem into simpler subproblems, the analysis of each of which (conceivably) provides new insights into the original problem as a whole. In this framework, the objective is to view the layout problem as a collection of subproblems; each subproblem should be efficiently solved and the solutions of the subproblems should be effectively combined.

In the two-step placement-routing approach the layout process is carried out as follows: The first step consists of finding a (good) placement of all the modules in the grid and the second step consists of interconnecting the terminals as specified by the nets. A well-known fact is the considerable difficulty of various subproblems introduced by this technique. Indeed, channel routing -- the simplest of the arising subproblems -- is intractable (NP-complete) [Sz] even in the powerful

knock-knee layout mode [Sa]. Thus, it is important to seek an alternative approach to the layout problem.

We propose a radically different technique for the layout problem -- a match-and-combine approach. The strategy is to *combine* a pair of strongly connected modules (which are called a *matched-pair*). Repeatedly applying this basic step produces a simple computation map (e.g. a map containing only one module) that can be trivially laid out. We shall elaborate on the new technique in the following sections.

Now we give a synopsis of this paper. Section 2 gives a formal description of the match-and-combine technique. In Section 3, a fundamental problem -- routing around a rectangle -- and variations of it, is discussed. Next, a problem essential to the method -- routing around two rectangles -- is analyzed. Finally, appropriate extensions are discussed.

2. Layout Algorithm

A circuit, represented as a computation map, is an interconnection of modules. The connections between two modules can be conceived as a "force" acting to bind them. Current layout methods either make use of this fact implicitly (e.g. partitioning techniques for placement) or adopt this idea, in the placement phase, when confronted with a pair of strongly connected modules (e.g. see [CHK,R]). We propose a systematic approach to the layout problem based on this intuitively sound principle.

In a more abstract formulation, a computation map can be represented by a weighted (hyper-) graph, called a *computation graph*, whose vertices correspond to the modules and whose edges correspond to the connections among the modules; the weight of an edge represents the number of connections in the associated computation map.

The intuitive formulation of the match-and-combine technique, to which we have alluded, is summarized as follows: Consider a computation graph in which two vertices v_i and v_j are strongly connected (the weight w_{ij} on the edge (v_i, v_j) is large) and each is weakly connected to other ver-

tices (the weights w_{ik} and w_{jk} on the edges (v_i, v_k) and (v_j, v_k), respectively, for $k \neq i,j$ are small with respect to w_{ij}), as shown in Figure 1. We can collapse (and thus, simplify) the previous computation graph into one with one less vertex by combining v_i and v_j into v_{ij}, or equivalently, by "pasting" M_i and M_j, in the corresponding computation map, to obtain M_{ij}. The two vertices v_i and v_j are said to be a *matched-pair*.

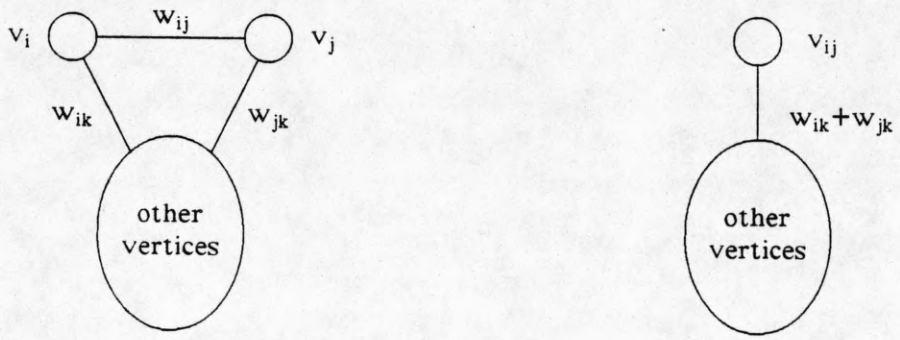


Figure 1. Simplifying a computation graph ($w_{ij} >> w_{ik}+w_{jk}$).

Remark: The qualitative condition ($w_{ij} >> w_{ik} + w_{jk}$) for combining two vertices can be expressed more formally; for instance, we could stipulate that v_i and v_j can be combined if $w_{ij} \geq \beta(w_{ik} + w_{jk})$, for a suitable $\beta > 0$.

The above step, is carried out as long as there exist matching pairs. When there remain no such pairs, we obtain a disconnected computation graph (if w_{ij} is below a threshold value Δ_{max} then v_i and v_j are treated as if they were disconnected) each of whose components is one of the following types:

a) *loose:* A vertex disconnected from the rest.

b) *tight:* A clique formed by a set of vertices, so that for any edge e connecting pairs of vertices of the clique we have weight(e) $\geq \Delta_{min}$ (a threshold to be selected).

Each tight component can be reduced to a loose one by successively matching and combining the vertices involved in the clique. This is an algorithmic formulation of the intuitive basis of the

technique. At some point there remains a collection of loose components, whose layout is trivial.

To efficiently apply the match-and-combine technique we must be able to understand and solve various problems involving one or two modules. These problems will be discussed in the following sections.

3. Routing Around One Module

The problem of routing around one module (R1M), both as a subproblem of the match-and-combine technique and as a basis for understanding more complex problems, is of central importance. We shall first analyze a simple version of R1M and then describe extensions thereof. Hereafter, we consider routing of two-terminal nets.

The problem of routing around a rectangle (R1R) is formulated as follows: Given a rectangle $\mathbb{R}$, interconnect the terminals on the boundary of $\mathbb{R}$ in the *layout domain* (i.e. the grid external to $\mathbb{R}$) as specified by a set of (two-terminal) nets.

The layout area A is the area of the smallest grid rectangle enclosing the whole layout. The goal is to find a layout with the minimum-area enclosing rectangle. Denoting by n the total number of terminals, an $O(n^3)$ time algorithm was proposed by LaPaugh [L] and, more recently, Gonzalez and Lee [GL] have presented an efficient but very complex algorithm, running in O(nlogn) time.

We shall now present a very simple and fast technique (also asymptotically running in O(nlogn) time) to solve an arbitrary instance of R1R. In addition, the algorithm is extensible to solve a number of related problems (e.g., the problem of *routing around two rectangles*), as we shall demonstrate.

The four corners of the rectangle $\mathbb{R}$ are labeled NW, NE, SW, and SE and the four sides are labeled T,B,L, and R, with obvious meanings as illustrated in Figure 2. Each net is classified as an S_1S_2-net with $S_1,S_2 \in \{T,B,L,R\}$, where S_1 and S_2 are two (not necessary distinct) sides of $\mathbb{R}$.

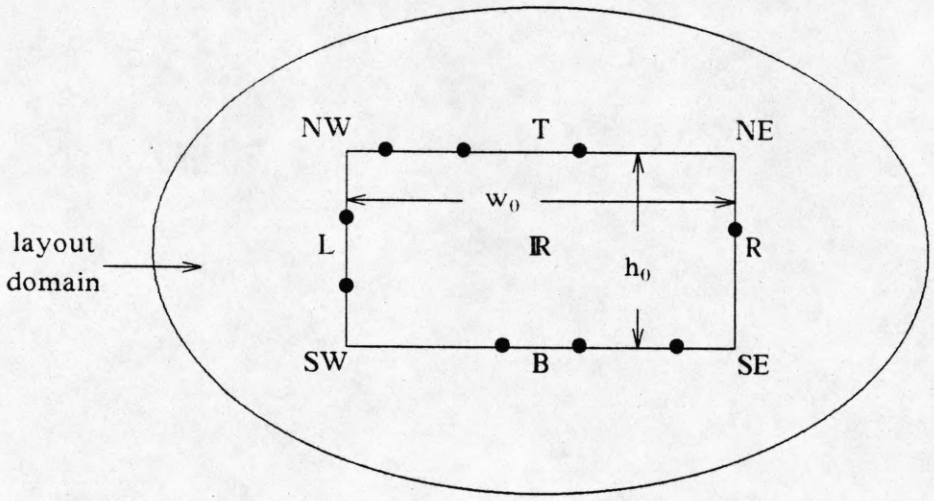


Figure 2. Routing Around a Rectangle.

Let $(h + h_0)$ and $(w + w_0)$ be the height and the width, respectively, of the grid rectangle enclosing the layout, where h_0 and w_0 are respectively the height and the width of IR. Thus, the goal is to minimize the area $A = (h+h_0)\,(w+w_0)$.

The first terminal of a (two-terminal) net N_j is the one closest to the SW corner in a clockwise scanning of the boundary of IR, starting from SW. As discussed in [L], it suffice to assign a direction to each net N_j : a clockwise direction, D(j) = CW or a counterclockwise direction, D(j) = CCW. The direction is defined by the wire realizing the net from its first terminal to its second (other) terminal. Once the directions are known for all nets, the actual layout can be obtained by employing a channel assignment algorithm (e.g., see [HS]).

A net with both terminals on same side or on adjacent sides of IR is called a *local net*; otherwise, a net (with both terminals on opposite sides of IR) is called a *global net*. The following is due to LaPaugh and can be easily proved.

Theorem 1 [L] : There exists an optimal layout with all the local nets having a *minimal-corner assignment*, that is, an assignment passing around the minimum number of corners of IR.

In conformity with our definition of first and second terminal of a net, the minimal-corner assignment corresponds to assigning a CCW direction to LB-nets and a CW direction to the rest of the local nets.

Thus, it remains to consider the global nets: the top-bottom TB-nets and the left-right LR-nets. Each TB-net increases the width w (and each LR-net increases the height h) by exactly one unit independently of the direction assigned to it; thus, we can process TB- and LR-nets independently, since the routing of a TB-net does not affect the final width (and similarly the routing of an LR-net does not affect the final height). Next, we describe an algorithm to optimally lay out TB-nets. It is clear, due to the symmetry, that exactly the same algorithm can be employed to optimally lay out LR-nets.

We focus our attention on the top (T) and the bottom (B) of $\mathbb{R}$ and consider only nets with at least one terminal on T or B. The columns (of the unit grid) are numbered left-to-right from 0 to c_t, at any stage of the execution of the algorithm. The maximum number of nets which cross at any point of the open interval $(c,c+l)$ and have already been assigned a direction is called the *used capacity* of $(c,c+l)$ and is denoted by $u(c,c+l)$, for $0 \leq c < c+l \leq c_t$. To simplify the notation $u(c)$ denotes $u(c,c+1)$.

Each TB-net, regardless of its direction, must pass around exactly two corners of $\mathbb{R}$; in this sense, the two directions (CW and CCW) are symmetric, that is, one is not preferred over the other. A closer look at each TB-net reveals an asymmetry, however, which suggests the assignment of a direction. Intuitively, a *barrier* is an open interval $(c,c+l)$ of high density, that is, we have both $u(c-1) < u(c,c+l)$ and $u(c+l) < u(c,c+l)$. If the assignment of a direction to a net causes this net to cross a large barrier, then a high price (in terms of the final height) must be paid.

More formally, for each net N_j, at any stage of the execution of the algorithm (on T or on B) we define the left barrier as the pair $(lb(j),L(j))$, where:

$$lb(j) = u(0,t_j) \text{ and } L(j) = \max\{c \mid u(c-1) = lb(j) \text{ and } c \leq t_j\} .$$

where t_j is the column containing the terminal of N_j on T†. The right barrier (rb(j),R(j)) for N_j is analogously defined (see Figure 3):

$$rb(j) = u(t_j,c_t) \text{ and } R(j) = \min\{c|u(c) = rb(j) \text{ and } t_j \leq c\}.$$

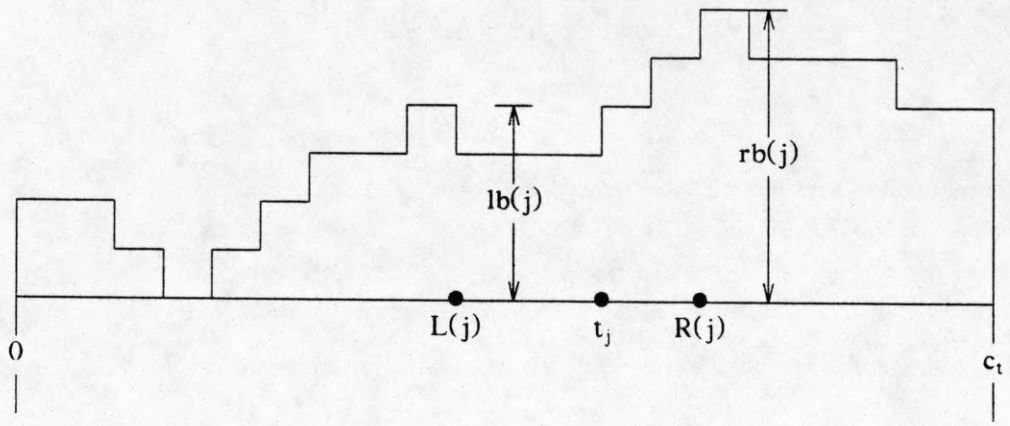


Figure 3. The left and the right barriers of N_j.

A net N_j with $lb(j) \neq rb(j)$ (either on T or on B) is called an *unstable* net. Without loss of generality consider the set of yet unassigned nets $\eta = \{N_{j_1}, \ldots, N_{j_k}\}$ on T with $lb(j_i) < rb(j_i)$ (i=1,....k). Clearly, all members of η have a common right barrier but they do not necessarily have identical left barriers. We call *balancing* the following task:

1. Select $N_s \in \eta$ with $b_s \leq b_{j_i}$ (i = 1,....k).

2. Assign the CCW direction to N_s, that is, D(s) = CCW.

3. Update the barriers after the assignment of a direction to N_s.

The balancing task is repeated until there are no more unstable nets‡. We refer to the layout produced by the balancing task as the *greedy layout*. For brevity, we say that a net N has been *assigned* to mean that N has been assigned a direction.

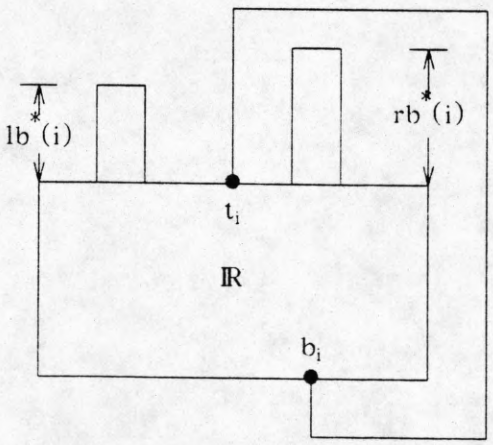
Lemma 1: The assignment of the unstable nets in the greedy layout is optimal.

† Similarly, if N_j has a terminal on B, this terminal occurs in column b_j.

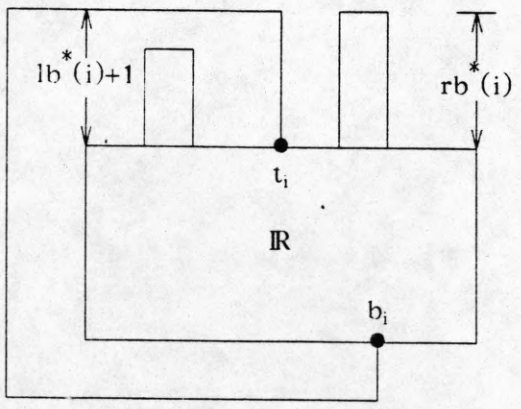
‡ Note that an initially unstable net may change its status as an effect of the balancing task.

Proof: Consider an optimal layout of $\mathbb{R}$. In this layout let N_i be the ith net to which a direction D(i) has been assigned in the balancing phase and let $D^*(i)$ be its direction in the given optimal layout, for $1 \leqslant i \leqslant k$. Assume the assignments of $N_1, \ldots, N_{i-1}$ in the optimal layout and in the greedy layout agree; this is certainly true, by virtue of Theorem 1, for i = 1. Our objective is to modify the optimal layout, without losing optimality, by changing the assignment of N_i to make it agree with its assignment in the greedy layout. In this process, as we will see, we may have to change the assignment of another net N_j such that $j > i$ (this inequality is *crucial*). We remove the wire, connecting the two terminals of N_i, from the optimal layout. We use the superscript * to denote the variables of this layout, e.g., $u^*(c,c+l)$ denotes the used capacity of the open interval $(c,c+l)$ in this layout. Without loss of generality let D(i) = CCW, that is, N_i has been given the CCW direction in the greedy layout. If $D^*(i) = D(i)$ then the claim holds. So assume $D^*(i) \neq D(i)$, that is, $D^*(i) = CW$.

Case 1) $lb^*(i) < rb^*(i)$: We change the direction of N_i, thereby decreasing the height on the top by one unit and (in the worst case) increasing the height on the bottom by one unit; thus the height of the new layout is not larger than the height of the optimal layout. The new layout is therefore optimal and the direction of N_i in this layout agrees with its direction in the greedy layout.



An optimal layout of $\mathbb{R}$



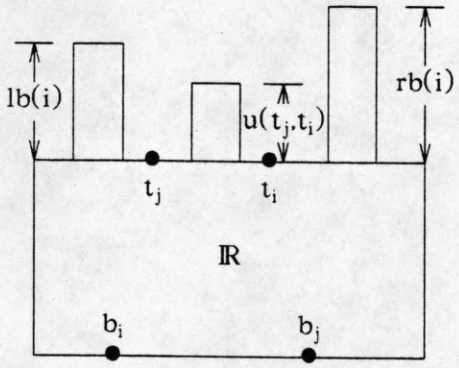
A new optimal layout

Case 2) $lb^*(i) \geq rb^*(i)$: After the completion of the (i-1)th step of the balancing phase (right before N_i is assigned) let $\eta_i = \{N_{i_1}, \ldots, N_{i_k}\}$ be the set of yet unassigned nets on T with $lb(i_j) < rb(i_j)$ for j = 1,....k. Clearly $N_i \in \eta_i$ and $b_i \leq b_{i_j}$ for j = 1,....k. We will show that there exists another net N_j (with j > i, that is, a net to be assigned later by the greedy algorithm) contributing only to $lb^*(i) - lb(i)$ and not to $u^*(R(i)) - u(R(i))$.

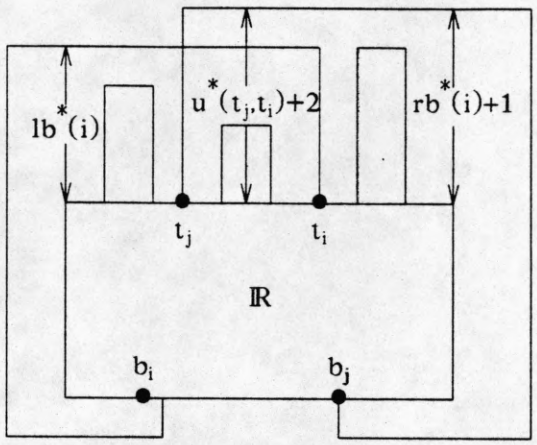
Since $lb^*(i) \geq rb^*(i) \geq u^*(R(i))$ and $lb(i) < rb(i) = u(R(i))$ then there are more nets (assigned after N_i) contributing to $lb^*(i) - lb(i)$ than there are nets contributing to $u^*(R(i)) - u(R(i))$. Clearly all these nets belong to η_i since their terminals on T is to the left of R(i). Let N_j be one of these nets with the rightmost terminal on T. Since $N_j \in \eta_i$ then $b_i < b_j$ (a decision made in the balancing phase). We claim that changing the directions of N_i and N_j, and thus making the direction of N_i to agree with the one assigned in the balancing phase, does not increase the height. Since $b_i < b_j$, the height on the bottom does not increase (in fact, it may decrease). On the top, one of the following two subcases may occur.

Subcase 2a) $t_i < t_j$: Clearly the height does not increase on the top. In fact all the used capacities remain the same and the used capacity of (t_i,t_j) on T and the used capacity of (b_i,b_j) on B decrease.

Subcase 2b) $t_i > t_j$: Changing the direction of N_i and N_j increases $u^*(t_j,t_i)$ by two units. We must guarantee $rb^*(i) + 1 \geq u^*(t_j,t_i) + 2$.



Before the balancing phase



Changing the direction of N_i and N_j

Clearly $rb(i) > u(t_j,t_i)$ (otherwise, $rb(i) \leqslant lb(i)$). All the nets that contribute to $u^*(t_j,t_i) - u(t_j,t_i)$ also contribute to $rb^*(i) - rb(i)$ because N_j was chosen to be the net (in $\eta_i - N_i$) with the rightmost terminal on T. So $rb^*(i) - rb(i) \geqslant u^*(t_j,t_i) - u(t_j,t_i)$ and thus $rb^*(i) \geqslant u^*(t_j,t_i) + 1$. We conclude $rb^*(i) + 1 \geqslant u^*(t_j,t_i) + 2$. □

Each unassigned net N_j at the completion of the balancing phase is called a *stable* net, that is, $rb(j) = lb(j)$: assume, there are s stable nets. These nets increase the final height by at least s units. The stable nets can be optimally laid out, employing the *pair-up* technique devised by Baker[B]. The strategy is to match the nets (N_i and N_j are matched if $t_i < t_j$ and $b_i < b_j$). A pair of matched nets increases the height by exactly 2 units (one unit on T and one unit on B). If s_1 matched pairs are found then $s_2 = s - 2s_1$ nets must pairwise cross, that is, denoting them as $N_1, \ldots, N_{s_2}$, we must have $t_1 < \cdots < t_{s_2}$ and $b_1 > \cdots > b_{s_2}$. These nets can be laid out using at most $s_2 + 1$ tracks, and thus, the contribution of stable nets is at most $2(s_1) + s_2 = s + 1$, and, it is easy to construct examples that require s + 1 unite (e.g., s nets that cross). The preceding discussion gives us the following result.

Lemma 2: The pair-up technique produces an optimal layout of stable nets.

We now give a formal description of the layout algorithms.

```
procedure LAYOUT (TB);
    begin while there exists an unstable net do
        begin if there exist a net N_j with lb(j) < rb(j) on S then        (* S ∈ {T,B}*)
            begin η:= set of nets in [0 , R(j)];
                N_s: = a member of η with the leftmost terminal on S̄;
                                        (* if S = T then S̄ = B and vice versa *)
                D(s) = CCW
            end
        else begin N_j := a net with lb(j) > rb(j) on S;
                η:= set of nets in [L(j) , c_t];
                N_s: = a member of η with the rightmost terminal on S̄;
                D(s) = CW
            end
        end
        if there exists any stable net then PAIRUP
    end.
```

Thus, an arbitrary instance of R1R can be laid out as follows:

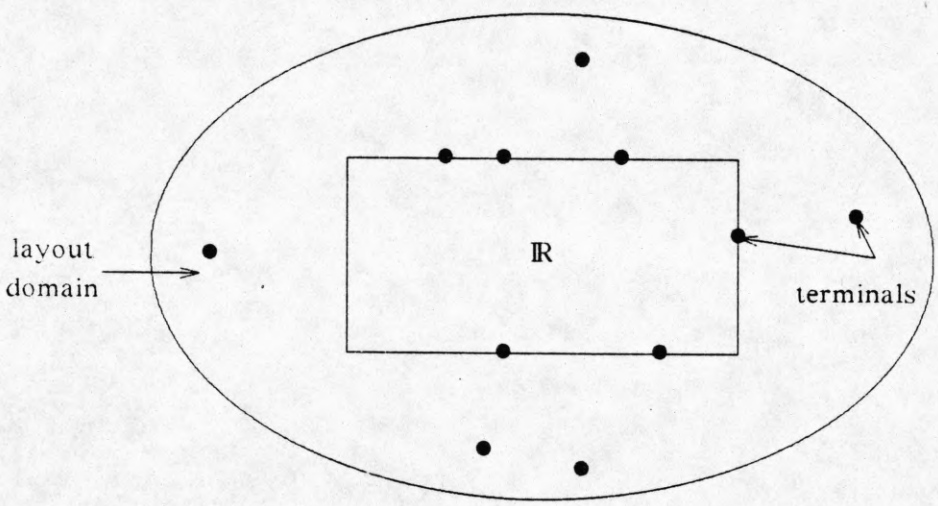
```
procedure LAYOUT-RECT;
    begin ASGN (local nets);          (*minimal-corner assignment *)
        LAYOUT (TB);                  (*balancing and
        LAYOUT (LR);                    pairing *)
        CHANNEL-ASGN;                 (*algorithm of [HS]*)
    end.
```

Theorem 2: LAYOUT-RECT produces an optimal layout for an arbitrary instance of R1R in O(nlogn) time, where n is the number of terminals.

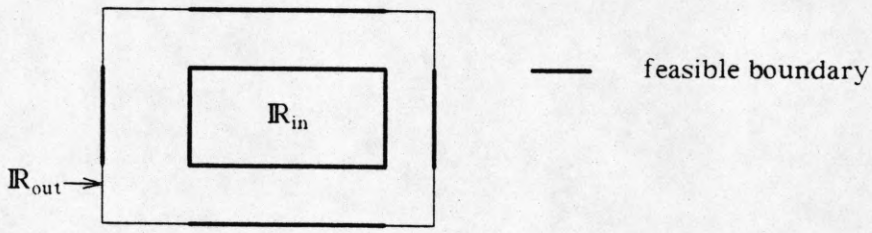
Proof: From Theorem 1, Lemma 1, and Lemma 2 we obtain that the layout produced by LAYOUT-RECT is optimal. Minimal-corner assignment and balancing cost one unit of time per net, assuming that the terminals are presorted (in O(nlogn) time). It is easy to see that PAIRUP takes O(nlogn) time [B]. □

We now introduce three extensions of R1R. The preceding algorithm can be easily modified to solve these problems in O(nlogn) time; moreover, it can be shown that the layouts are two- or three-layer wirable.

Problem 1: Given a rectangle $\mathbb{R}$, interconnect a set of terminals, placed anywhere in the layout domain external to $\mathbb{R}$ but including the boundary of $\mathbb{R}$, as specified by a set of (two-terminal) nets. At least one terminal of each net is on the boundary of $\mathbb{R}$.

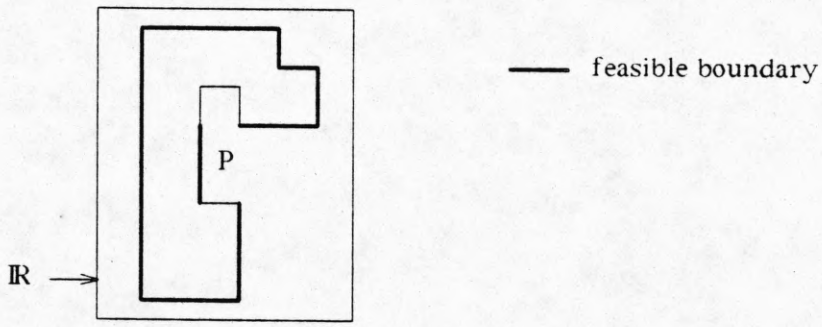


Problem 2: Given two isothetic rectangles $\mathbb{R}_{in}$ and $\mathbb{R}_{out}$, where $\mathbb{R}_{out}$ encloses $\mathbb{R}_{in}$, interconnect a set of terminals, located on the "feasible" boundaries (as shown below). The feasible boundary are the set of points on the boundary of $\mathbb{R}_{in}$ and $\mathbb{R}_{out}$ that can be connected to at least one point in $\mathbb{R}_{in}$ by means of a vertical or a horizontal line segment.



This problem was also solved by Suzuki-Ishiguro-Nishizeki [SIN] by viewing the problem as a multicommodity flow; it is not clear how many conducting layers are needed to wire the layouts produced by their technique (probably four).

Problem 3: Given a polygon P, interconnect a set of terminals located on the "feasible" boundary. The feasible boundary consists of the set of points on the boundary of P that can be connected, by means of a vertical or a horizontal line segment, to the boundary of a grid rectangle $\mathbb{R}$ external to P without crossing the boundary of P.



Remark: The entire boundary of the modules arising in practice (e.g., an L-shaped module), is feasible.

4. Routing Around Two Rectangles

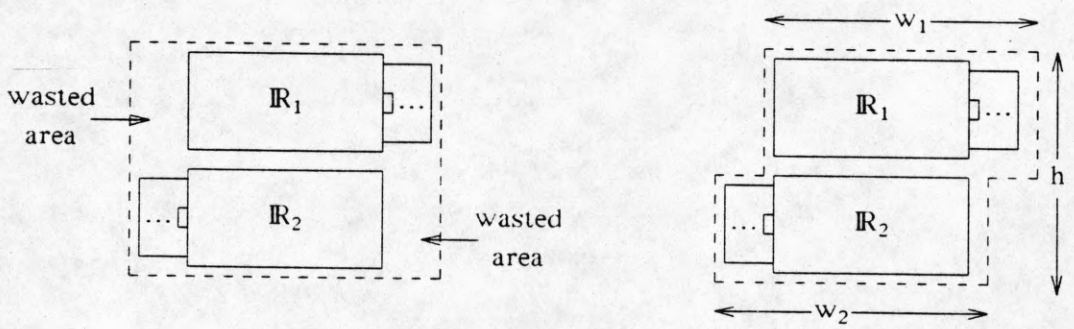
Problems involving two modules are of fundamental importance in the match-and-combine technique. Here, we analyze the simplest of all, which is easily extensible to more general prob-

lems.

The problem of routing around two rectangles (R2R) is formulated as follows: Given two vertically aligned rectangle $\mathbb{R}_1$ and $\mathbb{R}_2$ with equal widths (possibly different heights) interconnect a collection of terminals on their boundaries in the layout domain (the grid external to both $\mathbb{R}_1$ and $\mathbb{R}_2$) as specified by a set of (two-terminal) nets; the two rectangles can slide vertically with respect to each other ($\mathbb{R}_1$ is conventionally above $\mathbb{R}_2$).

For a restricted case of R2R, Chandrasekhar and Breuer [CB] have devised an algorithm that produces an optimal-area layout when certain types of nets are excluded. Recently, Baker [B] presented a 1.9 approximation algorithm with the goal of minimizing the perimeter of the final layout. Both techniques adopt the Manhattan layout mode.

Here, we present an algorithm for R2R which is a generalization of the technique described in Section 3. To reduce the wasted area, we shall aim to minimize the total height (h), the width of $\mathbb{R}_1$ (w_1) and the width of $\mathbb{R}_2$ (w_2) instead of minimizing the area of the enclosing rectangle, as shown in Figure 4. Doing so, produces $\sqcup\!\!\sqcap$-shaped modules having an area no more than that of the minimum-area enclosing rectangle. Thereafter, we use such shapes, instead of a rectangle, as a building block.



a. Minimizing the enclosing rectangle

b. Minimizing h, w_1, w_2

Figure 4

The sides of each rectangle are labeled as previously described, subscripted as the rectangle; for example L_2 means the left side of $\mathbb{R}_2$. Let $\mathbb{R}_{12}$ be an auxiliary rectangle enclosing $\mathbb{R}_1$ and $\mathbb{R}_2$ as

shown in Figure 5. First we project all the B_1 terminals on T_2 (or vice versa); at the completion of the assignment phase an optimal channel router (see [PL]) will be employed to lay out the channel between B_1 and T_2. A net with terminals on the same side or on the adjacent sides of $\mathbb{R}_1$, $\mathbb{R}_2$, or $\mathbb{R}_{12}$ is called a *local net*. L_1R_2-nets and L_2R_1-nets are called *cross nets*. The rest of the nets are called *global nets*.

Layouts of the local nets and the cross nets are trivial. The local nets are laid out using minimal-corner assignment (see Theorem 1) and the cross nets always pass through the center-channel (between B_1 and T_2): a simple variation of Theorem 1 verifies the optimality of this assignment. It remains to consider the global nets. L_1R_1- and L_2R_2-nets are laid out by employing LAYOUT(TB), described in Section 3. The rest of the global nets (T_1T_2,T_1B_2,T_2B_2) are laid out by a modified version of the balancing and the pair-up technique, as we will describe next.

For convenience, we call the channel above T_1 the top-channel (T), the channel between B_1 and T_2 the middle-channel (M), and the channel below B_2 the bottom channel (B). To simplify the notation, we refer to T_1T_2-nets, T_2B_2-nets, and T_1B_2-nets as TM-nets, MB nets, and TB-nets, respectively, as shown in Figure 6.

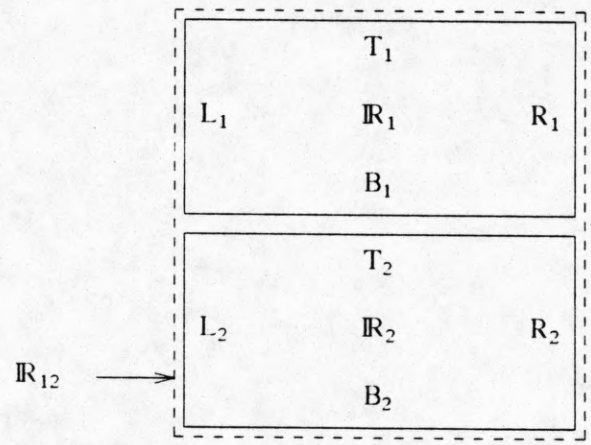


Figure 5. Routing Around Two Rectangles

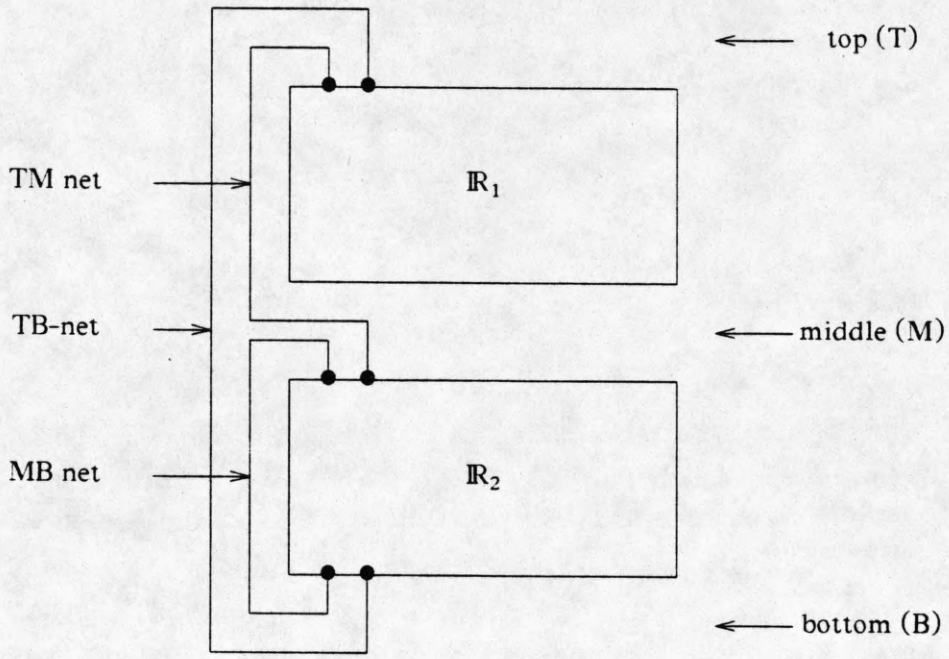


Figure 6. Labeling the Global Nets

From an optimal layout in which a TB-net N_i passes through M, we can obtain a different (but still optimal) layout in which N_i does not pass through M. If a net N_i has a terminal on T,M, or B, this terminal occurs at column t_i,m_i, or b_i, respectively.

Consider a net N_j on T with rb(j) > lb(j). Let $\eta_{TB} = \{N_{j_1},....,N_{j_k}\}$ and $\eta_{TM} = \{N_{j_{k+1}},....,N_{j_m}\}$ be respectively the set of TB and the set of TM unassigned nets in [0 , R(j)]. The nets in η_{TB} are called *unstable* if one of the following two conditions is satisfied (symmetric cases are omitted).

1) $|\eta_{TB}| > |\eta_{TM}|$

2) $|\eta_{TB}| = |\eta_{TM}|$ and rb(j) > lb(j)+1

We call *m-balancing* the following task:

1) Select $N_s \in \eta_{TB}$ with $b_s \leqslant b_{j_i}$ (i = 1,...,k).

2) Assign a CCW direction to N_s, that is, D(s) = CCW.

3) Update the barriers.

The m-balancing task is repeated until there are no more unstable nets. We refer to the layout produced by the balancing task the *greedy layout*.

Lemma 3. The m-balancing task produces an optimal layout of unstable nets.

Proof: As in the proof of Lemma 1, consider an optimal layout of $\mathbb{R}$. In this layout let N_i be the ith net to which a direction D(i) has been assigned in the balancing phase and let $D^*(i)$ be its direction in the given optimal layout, for $1 \leq i \leq k$. Assume the assignments of $N_1, \ldots, N_{i-1}$ in the optimal layout and in the greedy layout agree; this is certainly true, by virtue of Theorem 1, for $i = 1$. Our objective is to modify the optimal layout, without losing optimality, by changing the assignment of N_i to make it agree with its assignment in the greedy layout. In this process, as we will see, we may have to change the assignment of another net N_j such that $j > i$ We remove the wire, connecting the two terminals of N_i, from the optimal layout. We use the superscript * to denote the variables of this layout, e.g., $u^*(c,c+l)$ denotes the used capacity of the open interval $(c,c+l)$ in this layout. Without loss of generality let N_i be a TB-net and assume D(i) = CCW, that is, N_i has been given the CCW direction in the greedy layout. If $D^*(i) = D(i)$ then the claim holds. So assume $D^*(i) \neq D(i)$, that is, $D^*(i) = CW$.

Clearly $lb^*(i) \geq rb^*(i)$, otherwise, the result follows by virtue of Lemma 1 (case 1). Recall that N_i was assigned if one of the following two conditions was met:

Case 1) $|\eta_{TB}| > |\eta_{TM}|$:

Since $lb^*(i) \geq rb^*(i) \geq u^*(R(i))$ and $lb(i) < rb(i) = u(R(i))$ then there are more nets in $\eta_{TB} \bigcup \eta_{TM}$ (assigned after N_i) contributing to $lb^*(i) - lb(i)$ than there are nets contributing to $u^*(R(i)) - u(R(i))$. Since $\eta_{TB} > \eta_{TM}$ then at least one of these nets must be a TB-net. Let N_j be one of these TB-nets with the rightmost terminal on T. Since $N_j \in \eta_{TB}$ then $b_i < b_j$ (a decision made in the m-balancing phase).

Case 2) $|\eta_{TB}| = |\eta_{TM}|$ and $rb(i) > lb(i)+1$:

Since $lb^*(i) \geq rb^*(i) \geq u^*(R(i))$ and $lb(i)+1 < rb(i) = u(R(i))$ then there are (at least) two more nets (assigned after N_i) contributing to $lb^*(i) - lb(i)$ than there are nets contributing to

$u^*(R(i)) - u(R(i))$. Since $\eta_{TB} = \eta_{TM}$ then at least one of these nets must be a TB-net. Let N_j be one of these TB-nets with the rightmost terminal on T. Since $N_j \in \eta_{TB}$ then $b_i < b_j$ (a decision made in the m-balancing phase).

As it was proved in Case 2 of Lemma 1, we can change the direction of N_i and N_j while maintaining optimality. Thus the greedy layout is optimal. □

Each unassigned not N_j at the completion of the m-balancing task is called a stable net and must satisfy: lb(j) = rb(j), lb(j) = rb(j)+1, or lb(j) = rb(j)−1. Assume, there are s stable nets. These nets increase the final height by at least s − 3 units. We can employ the pair-up technique to lay out stable TB-, TM-, and MB-nets independently. As discussed before, the total height is increased by s units. Thus, the final layout is within three units from the optimal; in fact, we can construct examples (a crossing as discussed in Section 3) in which these three units are required. This means the pair-up technique is existentially optimal.

Lemma 4: The pair-up phase produces an optimal layout of stable nets.

We refer to the m-balancing and pair-up of TB-, TM-, and MB-nets as m-LAYOUT. An arbitrary instance of R2R can be laid out as follows:

```
procedure LAYOUT-2RECT;
    begin ASGN (local nets);              (*minimal-corner assignment *)
          ASGN (cross nets);              (* middle-center assignment *)
          LAYOUT (L1R1);                  (* see Section 3 *)
          LAYOUT (L2R2);                  (* see Section 3 *)
          m-LAYOUT (TB,TM,MB);            (* as described in Lemma 3*)
          CHANNEL-ROUTER;                 (* algorithm of [PL] *)
    end
```

Theorem 3: LAYOUT-2RECT produces an optimal layout for an arbitrary instance of R2R in O(nlogn) time, where n is the number of terminals.

Proof: From Theorem 2, Lemma 3, and Lemma 4 we obtain that the layout produced by LAYOUT-2RECT is optimal. As discussed in Theorem 2, minimal corner assignment and LAYOUT algorithms can be done in O(nlogn) time. Assignment of cross nets and mLAYOUT cost one unit of time per net; total of O(n) time. The algorithm of Preparata-Lipski used in the last step runs in O(n) time [PL]. □

The preceding algorithm can be easily modified to solve extended versions of R2R. These extensions correspond to the problems discussed at the end of Section 3.

References

[B] B. S. Baker, "A provably good algorithm for the two module routing problem," *SIAM Journal on Computing*, Vol. 15, n. 1, February 1986, pp. 162-188.

[CB] M. S. Chandrasekhar and M. A. Breuer, "Optimum placement of two rectangular blocks," Technical Report, Department of Electrical Engineering, University of Southern California.

[CHK] N. P. Chen, C. P. Hsu, and E. S. Kuh, "The Berkeley building block layout system," Technical Report, Electronics Research Laboratory, University of California, Berkeley.

[GL] T. F. Gonzalez and S. L. Lee, "An optimal algorithm for optimal routing around a rectangle," *Proceedings of the 20th Allerton Conference on Communication, Control and* Computing, Allerton, October 1982, pp 636-645.

[HS] A. Hashimoto and J. Stevens, "Wire routing by channel assignment within large apertures," *Proceedings of the 18th Design Automation Workshop*, June 1971, pp. 155-169.

[L] A. S. LaPaugh, "A polynomial time algorithm for optimal routing around a rectangle," *Proceedings of the 21st Symposium on Foundations of Computer Science, Syracuse*, October 1983, pp. 282-293.

[PL] F. P. Preparata and W. Lipski, "Optimal three-layer channel routing," *IEEE Transactions on Computers, C-33*, n.5, May 1984, pp. 427-437.

[R] R. L. Rivest, "The PI (Placement and Interconnection) system," *Proceedings of the 19th Design Automation Conference*, Los Vegas, June 1982, pp. 475-481.

[Sa] M. Sarrafzadeh, "On the complexity of the general routing problem in the knock-knee mode," *Proceedings of the twentieth annual conference on information sciences and systems*, Princeton NJ, March 1986.

[Sz] T. G. Szymanski, "Dogleg channel routing is NP-complete," *IEEE Transactions on CAD*, Vol. 4, n.1, January 1985, pp. 31-40.

[SIN] H. Suzuki, A. Ishiguro, and T. Nishizeki, "Routing in a region bounded by nested rectangles", manuscript.