KASAMI

of the universal relation model,

nulls? Proc. ACM SIGMOD
p. 108–120.
ir correct access paths, ACM

base management, TR CSRG-

SIAM J. COMPUT.
Vol. 14, No. 2, May 1985

RECTILINEAR GRAPHS AND THEIR EMBEDDINGS*

GOPALAKRISHNAN VIJAYAN† AND AVI WIGDERSON‡

Abstract. The embedding problem for a class of graphs called rectilinear graphs is discussed. These graphs have applications in many VLSI Layout Problems. An interesting topological characterization of these graphs lead to efficient algorithms for recognizing and embedding rectilinear graphs which are embeddable on the plane.

Key words. graphs, embeddings of graphs, VLSI layouts, algorithms for graph embedding

1. Introduction. The problem we address in this paper is an embedding problem for a class of graphs which we call rectilinear graphs. These graphs are important in many VLSI layout problems. In fact, this problem arose in the implementation of ALI [7], [8], a procedural language for VLSI design currently under development at Princeton. An embedding algorithm can be used to automate the production of VLSI layouts in many procedural design systems.

The following is an informal description of rectlinear graphs and their embeddings. The vertices of a rectilinear graph have degree at most four. The edges incident on each vertex are given distinct labels from the set {Left, Right, Up, Down}. Suppose we place the vertices on the grid points of a rectangular grid, and for each edge draw a straight line segment between its endpoints. We call the result an embedding of the graph, if the edges lie along grid lines, no two edges cross, and the directions of the edges at each vertex are consistent with their labels.

Consider the following model for VLSI layout design. A VLSI layout is described hierarchically using cells and wires that connect the cells together. Each cell C is enclosed within a rectangle $R(C)$, and has four lists of pins, one each for the left, top, right, and bottom of rectangle $R(C)$. Each wire w is denoted by a pair of pins (p_i, p_j), such that p_i and p_j are pins of different rectangles, and are of opposite types. For example, if p_i is a right pin then p_j should be a left pin. Given such a description of a VLSI layout, our aim is to produce an embedding of the description on the plane, such that (i) no two bounding rectangles touch each other, (ii) the pins appear in the correct order on the bounding rectangles, (iii) the wires are straight and rectilinear, and (iv) no two wires cross each other. Later on, we can fill each bounding rectangle $R(C)$ with the embedding of the cell C in the same manner.

The restriction that wires cannot be bent may seem unrealistic, but this is certainly the case in many design systems including ALI. If a wire has to be bent, the user specifies that by breaking up the wire into several straight wires and placing cells at each of the turn points of the wire. In ALI, for example, the user can incorporate routing algorithms in a ALI program to determine how the wires are to be bent. The restriction that wires cannot cross implies that we are dealing with the wires on a single layer. For a layout with multiple layers, it is clearly necessary that the wires on each layer do not cross.

It is easy to observe that the above description of a layout induces a rectilinear graph, whose vertices are the pins and the corners of the bounding rectangles, and

* Received by the editors December 22, 1982, and in revised form December 6, 1983.

† School of Information and Computer Science, Georgia Institute of Technology, Atlanta, Georgia 30332. The research of this author was supported in part by DARPA under grant N0014-82-K-0549.

‡ Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, California 94720. The research of this author was supported in part by an IBM fellowship.

whose edges are the wires and the segments created on the bounding rectangles by the vertices.

For VLSI applications, we need efficient algorithms to recognize and then actually embed rectilinear graphs. In this paper, we present an $O(n)$ recognition algorithm and an $O(n^2)$ embedding algorithm, where n is the number of vertices in the graph. Thus, a hierarchically described VLSI layout with cell instances $C_1, C_2, \cdots, C_m$ can embedded in time $O(\sum_{i=1}^{m} n_i^2)$, where n_i is the number of pins in cell instance C_i.

An embedding of a rectilinear graph is just a relative placement of the vertices (cells) on a rectangular grid, such that no two edges cross. Some of the relative placement information is already present in the description of a rectilinear graph. For example, if (a, b) is the rightgoing edge of vertex a, then a should be to the left of b, and a, b should be on the same horizontal grid line. Hence, an embedding can be viewed as a "completion" of the rectlinear graph description. We showed in a different paper [10] that the completion problem for a slightly more relaxed VLSI layout model is NP-complete. In light of this result, the results in this paper have become more important.

In § 2, we present formal definitions of rectlinear graphs and thier embeddings. In § 3, we mention some properties of rectlinear graphs. We discuss some topological properties of the embeddings in § 4. A necessary and sufficient condition for biconnected rectilinear graphs to be embeddable is presented in § 5. A similar condition for arbitrary rectilinear graphs is the main result in § 6. We also describe a $O(n)$ recognition algorithm in this section. In § 7, we use the ideas of the previous sections to obtain an $O(n^2)$ embedding algorithm. An important subclass of rectilinear graphs is discussed in § 8. In § 9, we discuss extensions and open problems. For definitions of graph theoretic terminology used in this paper, please refer to [1], [2].

2. Definition of the problem. First we give a formal definition of a rectilinear graph.

DEFINITION 2.1. A *rectilinear graph* G is a triple (V, E, λ), where V is the vertex set, E is the edge set, and

$$\lambda : V \times V \to \Sigma \cup \{\varepsilon\}, \quad \text{where } \Sigma = \{L, R, D, U\}$$

is a *vertex ordering relation* with the following properties:

for every $a, b, c \in V$ and $X \in \Sigma$

(i) $\lambda((a, b)) = \varepsilon \Leftrightarrow \{a, b\} \notin E$
(ordering is specified only between adjacent vertices);

(ii) $\lambda((a, b)) = L \Leftrightarrow \lambda((b, a)) = R$, $\lambda((a, b)) = D \Leftrightarrow \lambda((b, a)) = U$;

(iii) $\lambda((a, b)) = X \Rightarrow \lambda((c, b)) \neq X, \forall c \neq a$ (no overlapping edges).

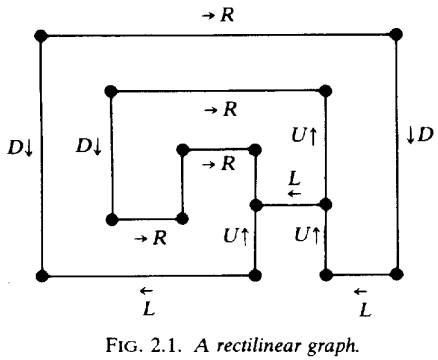


FIG. 2.1. *A rectilinear graph.*

Each vertex in a rectilinear graph has degree at most four, and each edge (a, b), as it goes from one vertex a to another b, has a nonempty label on it, which in the embedding will indicate the direction (left, right, down, or up) in which the edge leaves a vertex a. There can be at most one edge with a particular label emanating from each vertex. The undirected graph $G(V, E)$ will be referred to as the *underlying graph.* Figure 2.1 (like all other figures) gives an illustration of a rectilinear graph.

Now we define what sort of an embedding we are looking for.

DEFINITION 2.2. An *embedding* of a rectilinear graph $G(V, E, \lambda)$ on a *rectangular grid* is given by two mappings x, y: $V \to Z$ (the integers) which are the x and y coordinates respectively of the vertices. These mappings obey:

1. *The ordering relation,* λ, i.e. for all edges $\{a, b\} \in E$

$$\lambda((a, b)) = L \Rightarrow y(a) = y(b),\ x(a) > x(b),$$

$$\lambda((a, b)) = R \Rightarrow y(a) = y(b),\ x(a) < x(b),$$

$$\lambda((a, b)) = D \Rightarrow x(a) = x(b),\ y(a) > y(b),$$

$$\lambda((a, b)) = U \Rightarrow x(a) = x(b),\ y(a) < y(b).$$

2. *Planarity,* no two edges cross, i.e. for each pair of nonadjacent edges $\{a, b\}$, $\{c, d\}$ such that $\lambda((a, b)) = R$ and $\lambda((c, d)) = U$, the relation

$$x(a) \leqq x(c) \leqq x(b) \quad \text{and} \quad y(c) \leqq y(a) \leqq y(d)$$

does not hold.

An embedding of a rectilinear graph on a rectangular grid is one in which the vertices are placed at grid points, the edges run along grid lines in the directions given by their labels, and no two edges cross each other except if they share a vertex. Also, an edge cannot touch a vertex unless it is incident on it. We say that a rectilinear graph is embeddable if it has an embedding. We will show in the next section that not all rectilinear graphs are embeddable.

Now our main problem can be stated simply: Given a rectilinear graph $G(V, E, \lambda)$, is it embeddable, and if it is, find an embedding.

3. Some comments on rectlinear graphs. In this section we list some properties of rectilinear graphs and their embeddings. Some of these properties will give an indication of why our problem is different from other embedding problems, in particular, planar graph embedding [3], [6].

1. Embeddability is a hereditary property. Subgraphs are defined in the usual fashion, but here the labels of edges are inherited. This is obvious, but worth mentioning, because this will be used in the proofs.

2. If each connected component of a rectilinear graph is embeddable then the graph itself is embeddable. So, without loss of generality we will restrict ourselves to connected rectilinear graphs.

3. Rectilinear graphs with nonplanar underlying graphs are clearly not embeddable. So it is not interesting to consider those graphs. However, not every rectilinear graph with a planar underlying graph is embeddable. In Fig. 3.1, we have two simple cycles which are not embeddable.

4. In contrast with planarity, embeddability is *not* a property determined by the biconnected components. Fig. 3.2 provides an illustration of this fact.

5. This problem is a restriction of an NP-complete problem [10], [12]. For each wire w, we are given its orientation (horizontal or vertical), and a set V_w of vertices.

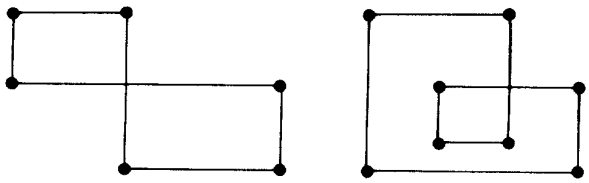


FIG. 3.1. *Two nonembeddable rectilinear cycles.*

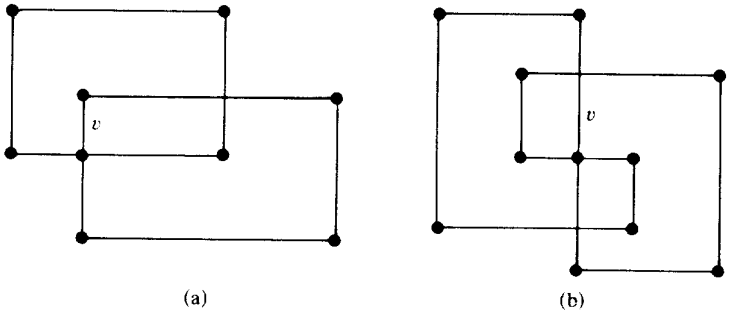


FIG. 3.2. *Two nonembeddable rectilinear graphs whose biconnected components are embeddable.*

The wire w has to touch each vertex in the set V_w (the vertices could be touched in any order). Then, the embedding problem becomes NP-complete.

6. If we relax the rectilinearity of the edges and impose only the cyclic ordering of the edges at each vertex, then there is an $O(|V|)$ algorithm [11]. The cyclic orderings automatically determines the faces of the embedding (if one exists). Thus a embeddable rectilinear graph has a unique embedding in this sense.

4. Topological structure of embeddings. There is a natural way to extend the function λ to paths and cycles in the graph as follows. Given a path $P = (v_0, v_1, \cdots, v_t)$ we define $\lambda(P) = \lambda((v_0, v_1))\lambda((v_1, v_2)) \cdots \lambda((v_{t-1}, v_t))$. We define a similar extension for cycles where now $v_t = v_0$. λ becomes a mapping that associates with each path or cycle in the graph a string in Σ^* which is the concatenation of labels along the path or cycle. Note that strings containing *RL*, *DU*, *LR*, *UD* as substrings do not represent paths. Also the direction in which we traverse a path and the starting point in a cycle are important. An example of this mapping can be found in Fig. 4.1.

Next we define two topological operations on rectilinear graphs. These operations will simplify a rectilinear graph while preserving its topological structure. Let G be a rectilinear graph.

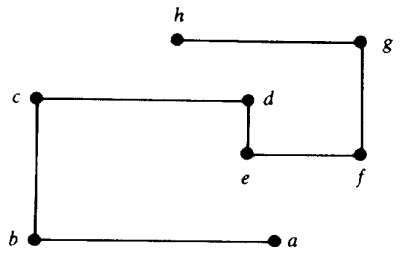


FIG. 4.1. *The extension of λ to paths.*

Operation 1—*Edge contraction.* Let $(abcd)$ be a path in G such that both b and c have degree 2, and $\lambda((abcd)) = XYX$ where $X, Y \in \Sigma$. Contract the edge (b, c) to the vertex b. The resulting path (abd) will have $\lambda((abd)) = XX$. We abbreviate this operation by $XYX \to XX$ (Fig. 4.2(1)).

Operation 2—*Vertex deletion.* Let (abc) be a path in G such that vertex b has degree 2, and $\lambda((abc)) = XX$ where $X \in \Sigma$. Delete the vertex b and introduce the edge (a, c). The resulting edge (a, c) will have $\lambda((a, c)) = X$. We abbreviate this operation by $XX \to X$ (Fig. 4.2(2)).

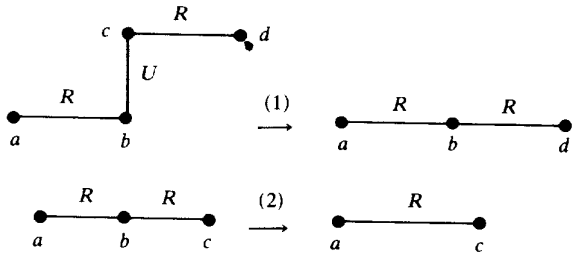


FIG. 4.2. *Edge contraction and vertex deletion.*

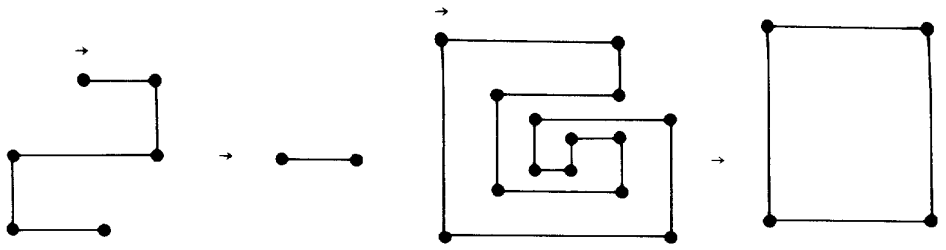
In a natural way we can define inverses for the above two operations which we will refer to as *edge expansion* and *vertex addition* respectively.

LEMMA 4.1. *Let G be a rectilinear graph and G' be the graph resulting from G by the application of a sequence of the above four operations. Then G' is also rectilinear and moreover G' is embeddable if and only if G is embeddable.*

Proof. The proof is easy and is left to the reader. □

DEFINITION 4.1. Given a string $\gamma \in \Sigma^*$ representing a path or a cycle, the *simplified form* $\bar{\gamma}$ *of* γ is obtained by repeatedly applying the reduction rules $XYX \to XX$ and $XX \to X$, where $X, Y \in \Sigma$ until they cannot be applied any more. If γ represents a cycle then it is treated as a cyclic string.

In Fig. 4.3 we give a path and a cycle along with their simplified forms.



$RDLDR \to R$ $\qquad$ $RDLDRULDLURDLU \to RDLU$

FIG. 4.3. *Simplification of a path and a cycle.*

LEMMA 4.2. *Every string $\gamma \in \Sigma^*$ has a unique simplified form.*

Proof. The replacement system defined by the two reduction rules have the Church–Rosser property [9]. □

DEFINITION 4.2. A *square* is one of the cyclic strings $LURD$ or $LDRU$.

Sometimes we may distinguish between two squares by their starting labels.

DEFINITION 4.3. A *spiral* is a substring of $(LURD)^+$ or $(LDRU)^+$.

LEMMA 4.3. *Every path is embeddable.*

Proof. Every spiral is embeddable. Since any path simplifies to a spiral, by Lemma 4.1 it is also embeddable. □

So it is the cycles which make the problem nontrivial. The following lemma is a crucial fact about cycles.

LEMMA 4.4. *A cycle is embeddable if and only if it simplifies to a square.*

Proof. If. A square is embeddable and hence by Lemma 4.1 any cycle which simplifies to a square is also embeddable.

Only if. Let f be an embeddable cycle and $\lambda(f)=\gamma$. By Lemma 4.1, the cycle defined by $\bar{\gamma}$ is also embeddable. Let $|\bar{\gamma}|=n$. Look at the embedding of $\bar{\gamma}$. Since it has no crossings the embedding is a simple polygon. Therefore the interior angles of this polygon sum to $(n-2)\times 180°$. Since $\bar{\gamma}$ is a spiral all its interior angles are 90°. The only solution to $n\times 90=(n-2)\times 180$ is $n=4$. Therefore $\bar{\gamma}$ is a square. □

The proof of the previous lemma suggests another useful characterization of embeddable cycles. Going along a cycle $f=v_1v_2\cdots v_nv_1$ in the counterclockwise direction, let us denote by $\varphi_f(v_i)$ the angle at vertex v_i, which is the angle between (v_{i-1}, v_i) and (v_i, v_{i+1}), and by $\varphi(f)$ the sum of these angles.

LEMMA 4.5. *A cycle $f=v_1v_2\cdots v_nv_1$, $n\geqq 4$ is embeddable if and only if* $\varphi(f)=\sum_{i=1}^{n}\varphi_f(v_i)=(n\pm 2)\times 180°$.

Proof. Suppose f is embeddable, then its embedding is a simple polygon. Depending on whether we sum the interior angles or exterior angles we should get $(n\pm 2)\times 180°$.

To prove the sufficient part we show by induction on n that f simplifies to a square. The possible values for $\varphi_f(v_i)$ are 90°, 180°, 270°. The basis for induction is $n=4$. In this case the given sum of the angles is either 360° or 1,080°, which implies that each angle is either 90° or 270° respectively. So f must be a square by itself.

Assume that the claim is true for all values less than n and let $n>4$. If for some i, $\varphi_f(v_i)=180°$ then $\lambda(v_{i-1}v_iv_{i+1})=XX$. We can apply vertex deletion at v_i to obtain $f'=v_1v_2\cdots v_{i-1}v_{i+1}\cdots v_nv_1$. Then $\varphi(f')=\varphi(f)-\varphi_f(v_i)=((n-1)\pm 2)\times 180°$, and by induction we are done.

This leaves the case where all angles are either 90° or 270°. Since $n>4$ and $\varphi(f)=(n\pm 2)\times 180°$ not all the angles can be equal. Hence there must be a k such that $\varphi_f(v_k)\neq\varphi_f(v_{k+1})$. Hence we have $\lambda(v_{k-1}v_kv_{k+1}v_{k+2})=XYX$. Apply edge contraction to obtain $f'=v_1\cdots v_{k-1}uv_{k+2}\cdots v_nv_1$. The edge contraction removed 360° from the angle sum and added 180°. Hence $\varphi(f')=((n-1)\pm 2)\times 180°$. □

DEFINITION 4.4. A *complement* of a path P *with respect to a square* σ is any path P^c in the graph such that PP^c is a cycle which simplifies to σ.

LEMMA 4.6. *Given a path P, all its complements with respect to a square σ, which have the same start and end labels, have a unique simplified form.*

Proof. Let $\lambda(\bar{P})=\alpha=X_1X_2\cdots X_k$. Since α is a spiral we have $X_i=X_j$ for $i=j(4)$. Assume that $k>4$ and that the spiral α and the square σ are either both clockwise or both counterclockwise. Then σ must be a substring of α. Since σ is a cyclic string we can assume that $\sigma=X_1X_2X_3X_4$.

Let P^c be a complement of P with respect to σ and let $\overline{\lambda(P^c)}=\beta$. Since $k>4$, β must spiral in the opposite direction to α. Since both α and β are simplified $\alpha\beta$ can be simplified only at the borders between the two strings. Write $\beta=\beta_1\beta_2\beta_3$, such that $\overline{\beta_3\alpha\beta_1}=\alpha$. We are allowed to shift β_3 because $\alpha\beta$ is a cyclic string. Then it is clear that $\beta_1\in\{X_{k-3}X_k, X_k, \varepsilon\}$ and $\beta_3\in\{\varepsilon, X_1, X_1X_4\}$. β_2 is the "essential part" of β. Since $|\alpha|=k$ and $\sigma=4$, we must have $|\beta_2|=k-4$. From the possible values of β_1 and β_3, and the fact that β is a spiral opposite in direction to α, we can conclude that $\beta_2=X_{k-1}X_{k-2}\cdots X_4$. We use $k>4$ in order for β_2 not to be an empty string. Therefore

$\beta = \{\varepsilon, X_k, X_{k-3}X_k\}X_{k-1} \cdots X_4\{\varepsilon, X_1, X_1X_4\}$, which is unique but for the start and end labels. The arguments in the cases where α and σ are in opposite directions and for $k \leqq 4$ are similar. □

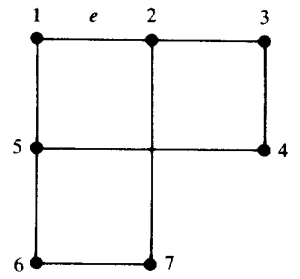
5. Biconnected rectilinear graphs. In this section we discuss an algorithm for recognizing biconnected rectilinear graphs. Note that the ordering relation λ induces a cyclic ordering of the edges incident at each vertex v. For convenience we will need the following definition.

DEFINITION 5.1. Let v be a vertex in a rectilinear graph G. Define $L_G(v)$ to be the cyclic list of the neighbors of v in G in the counterclockwise order.

Using these lists, we can define the essential notion of a candidate face of a biconnected rectilinear graph.

DEFINITION 5.2. Let $G = (V, E, \lambda)$ be a biconnected rectilinear graph. With each edge $e = (v_1, v_2)$, $v_1 > v_2$[1] we associate two lists of vertices called *candidate faces* $CF_1(e)$ and $CF_2(e)$ which are defined as follows. $CF_1(e) = v_1, v_2, \cdots, v_k, v_{k+1}$ where $v_i \neq v_j$ for $1 \leqq i < j \leqq k$, and $v_{k+1} = v_i$ for some i, $1 \leqq i < k-1$. Also, for each l, $1 < l < k+1$, v_{l+1} is the successor of v_{l-1} in the cyclic list $L_G(v_l)$. $CF_2(e)$ is similarly defined but starting with v_2, v_1.

It is easy to see that CF_1 and CF_2 are uniquely defined. An illustration of this definition is given in Fig. 5.1.



$L_G(1) = (5, 2)$ and $L_G(2) = (1, 7, 3)$
$CF_1(e) = 2, 1, 5, 6, 7, 2$ and $CF_2(e) = 1, 2, 7, 6, 5, 4, 3, 2$

FIG. 5.1. *Candidate faces.*

We now need a lemma about biconnected undirected graphs. Let us define a biconnected graph to be *minimal* if for every edge e in the graph $G-e$ is not biconnected. The following lemma is taken from [2] and is stated without proof.

LEMMA 5.1. *If G is a minimal biconnected graph having at least four vertices then G contains a vertex of degree two.*

LEMMA 5.2. *In any biconnected graph G which is not a simple cycle, there is a simple path $P = (v_1, v_2), (v_2, v_3), \cdots, (v_{r-1}, v_r)$, $r \geqq 2$, with the intermediate vertices (if any) v_i, $1 < i < r$ all having degree 2, such that the graph $G' = G - P$ is biconnected.*

Proof. Transform the given graph G to another graph G'' by replacing all paths of the form $P = (v_1, v_2), (v_2, v_3), \cdots, (v_{r-1}, v_r)$ where the vertices v_i, $i \neq 1, r$ all have degree 2, by the edge (v_1, v_r). So for each edge e in G'' we have a corresponding path P_e in G. Note that the degree of any vertex in G'' is at least three. If G'' has multiple edges between some two vertices, say u and w, then in G there must be at least two parallel paths between u and w. Since G is not a simple cycle any one of those paths

[1] For convenience we assume that V is a set of integers.

will serve our purpose. If G'' does not have multiple edges then it must have at least 4 vertices. By Lemma 5.1 G'' cannot be minimal. Therefore there is an edge e in G'' such that $G''-e$ is biconnected, which implies that $G-P_e$ is also biconnected. □

The following theorem gives a necessary and sufficient condition for a biconnected rectilinear graph to be embeddable.

THEOREM 5.1. *Let $G=(V, E, \lambda)$ be a biconnected rectilinear graph with at least three edges. Then G is embeddable if and only if for each edge e in the graph both the candidate faces $CF_1(e)$ and $CF_2(e)$ represent simple embeddable cycles in the graph (i.e. the starting and ending vertices are identical, and it simplifies to a square). Moreover, if the graph is embeddable each such distinct candidate face corresponds to a face in the planar embedding.*

Proof. Only if. Supposing for some edge $e=(v_1, v_2)$, $CF_1(e)$ is not a cycle, i.e. $CF_1(e)=v_1, v_2, \cdots, v_k, v_{k+1}$ with $v_i=v_{k+1}$ for some i, $1<i<k-1$. Suppose G is embeddable. Look at the cycle $v_i, \cdots, v_k, v_{k+1}$ in the embedding. Suppose that the edge (v_{i-1}, v_i) is inside this cycle. There can be no other edges (u, v_j), $i<j\leqq k$ *inside* this cycle, otherwise u would have appeared instead of v_{j+1} in $CF_1(e)$. From this observation and the fact that the embedding is planar, it follows that v_i is an articulation vertex, which contradicts the biconnectedness of G. The case where (v_{i-1}, v_i) is *outside* the cycle is similar (both cases are depicted in Fig. 5.2).

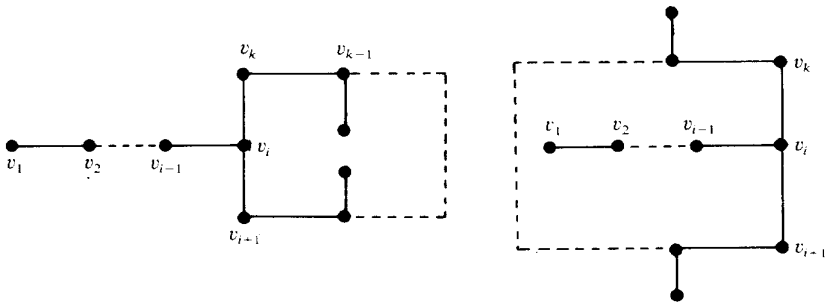


FIG. 5.2. *Two possible embeddings of $CF_1(e)$.*

Suppose $CF_1(e)$ is a cycle but is not embeddable. Since $CF_1(e)$ is a subgraph of G, G itself cannot be embeddable. Similar arguments hold for $CF_2(e)$.

If. The proof of this part is by induction on the number of edges. The basis for the induction are simple embeddable cycles. Assume that the claim is true for any biconnected rectilinear graph which has less than k edges. Let G be a biconnected rectilinear graph which is not a simple cycle and which has k edges. By Lemma 5.2, there is a simple path $P=(v_1, v_2), (v_2, v_3), \cdots, (v_{r-1}, v_r)$ with the vertices v_i, $i\neq 1, r$ all having degree 2, such that the graph $G'=G-P$ is biconnected. v_1 and v_r will have degree greater than two. Also assume that $v_1>v_2$ and $e_{12}=(v_1, v_2)$.

Since all our candidate faces are cycles, if an edge e lies on a candidate face f then either $CF_1(e)=f$ or $CF_2(e)=f$. So each edge will be present in exactly two of these candidate faces. Hence the path P will appear in $CF_1(e_{12})$ and its reverse path will appear in $CF_2(e_{12})$. Let

$$f_1=CF_1(e_{12})=v_1, v_2, \cdots, v_r, u_1, \cdots, u_j, v_1,$$

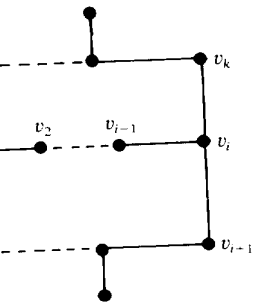
$$f_2=CF_2(e_{12})=v_r, v_{r-1}, \cdots, v_1, w_1, \cdots, w_k, v_r, \text{ and}$$

$$f_s=v_1, w_1, \cdots, w_k, v_r, u_1, \cdots, u_j, v_1.$$

then it must have at least
there is an edge e in G''
s also biconnected. □
ondition for a biconnected

tilinear graph with at least
dge e in the graph both the
ddable cycles in the graph
ifies to a square). Moreover,
ce corresponds to a face in

$CF_1(e)$ is not a cycle, i.e.
$< i < k-1$. Suppose G is
bedding. Suppose that the
dges (u, v_j), $i < j \leqq k$ *inside*
v_{j+1} in $CF_1(e)$. From this
ows that v_i is an articulation
se where (v_{i-1}, v_i) is *outside*



$_1(e)$.

ince $CF_1(e)$ is a subgraph of
old for $CF_2(e)$.
mber of edges. The basis for
nat the claim is true for any
ges. Let G be a biconnected
has k edges. By Lemma 5.2,
$_r$) with the vertices v_i, $i \neq 1, r$
connected. v_1 and v_r will have
$e_{12} = (v_1, v_2)$.
e e lies on a candidate face f
be present in exactly two of
$CF_1(e_{12})$ and its reverse path

$u_j, v_1,$
$\cdot, w_k, v_r,$ and

It follows from the definition of the candidate faces f_1 and f_2 that the vertices u_j, v_2, w_1 appear consecutively in that order in $L_G(v_1)$ and that w_k, v_{r-1}, u_1 appear similarly in $L_G(v_r)$ (see Fig. 5.3). Therefore for each edge in f_1 or f_2 which is not in P, the new candidate face in G' will be f_3 which is a simple cycle.

We still have to show that f_3 is embeddable. Since f_1 and f_2 are both embeddable $\varphi(f_1) = (r+j\pm 2)\times 180°$ and $\varphi(f_2) = (r+k\pm 2)\times 180°$. However, since f_1 and f_2 share the edge e_{12} it is implied by the definition of candidate faces that $\varphi(f_1) = (r+j+2)\times 180°$ and $\varphi(f_2) = (r+k+2)\times 180°$ is impossible. With a little bit of algebraic manipulation we can show that $\varphi(f_3) = ((j+k+2)\pm 2)\times 180°$. Since f_3 has $j+k+2$ vertices by Lemma 4.5, it is embeddable. Thus the candidate faces for G' are the same as those for G, excepting for f_3 replacing the two faces f_1 and f_2. So for each edge in G' its two candidate faces are again simple embeddable cycles. By the induction hypothesis G' is embeddable and each distinct candidate face corresponds to a face in its embedding. The orderings of the edges at the vertices v_1 and v_r imply that the end edges (v_1, v_2) and (v_{r-1}, v_r) of the path P are both trying to go inside the face corresponding to f_3.

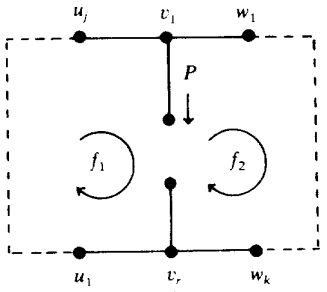


FIG. 5.3. *The two cycles f_1, f_2 and the path P.*

We are left to show that we can add the path P back without destroying embeddability. Find any rectilinear path P' in the face corresponding to f_3 in the embedding of G', that starts and ends with $\lambda((v_1, v_2))$ and $\lambda((v_{r-1}, v_r))$ respectively. This is clearly possible although we may have to extend the grid in order for P' to lie on the grid lines. P' creates a face in the embedding with the path $P_1 = v_r u_1 u_2 \cdots u_j v_1$. If f_3 is not the outside face then $\overline{\lambda(P_1P')} = \overline{\lambda(f_1)} = \overline{\lambda(P_1P)} = \sigma$. The case when f_3 is the outside face is slightly more complicated. There are two such different paths P' depending on the new outer face that is created. However, for one of the two the above holds and suppose this is the one we chose. By definition both P and P' are complements of P_1 with respect to σ, they also share the same start and end labels, and by lemma 4.6 we have $\overline{\lambda(P)} = \overline{\lambda(P')}$. Therefore $G'+P'$ can be obtained from G by applying a sequence of the four topological operations, and since $G'+P'$ is embeddable, by Lemma 4.1 G is also embeddable. It is easy to see that the two new faces we get after inserting P in the embedding of G' correspond to f_1 and f_2. □

The above theorem leads to the following algorithm for recognizing embeddable biconnected rectilinear graphs. The algorithm also outputs the faces of the embedding if the graph happens to be embeddable.

Algorithm *check-biconnected*(G);
begin
  if G is an edge **then** *return*;

```
if |E| > 3 |V| − 6 then
begin
  write ("not embeddable");
  quit
end;
for each edge e do
begin
  mark [e, 1] := false;
  mark [e, 2] := false
end;
for each edge e do
  for i := 1 to 2 do
  begin
    if not mark [e, i] then
    begin
      f := candidate-face (e, i);
      if not embed-cycle (f) then
      begin
        write ("not embeddable");
        quit
      end;
      for each edge e' = (v1, v2) in f do
        if v1 > v2 then mark [e', 1] := true
        else mark [e', 2] := true;
      output (f)
    end
  end
end.
```

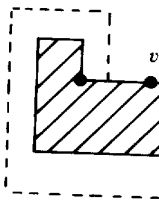
Boolean function *embed-cycle* (f) returns value *true* if f is an embeddable cycle. If f is a cycle then we simplify using the reduction rules and check if we end up with a square. This can be done in time linear in the size of f. Function call *candidate-face* (e, i) returns the candidate face $CF_i(e)$ and the function can be implemented exactly as described in Definition 5.2. In the calls to this function, each edge e can be traversed at most twice, due to the flags mark $[e, 1]$ and mark $[e, 2]$. Therefore the algorithm runs in time $O(|V|)$. We conclude this section with a lemma which will let us identify the outer face in a rectilinear graph.

LEMMA 5.3. *Let G be an embeddable biconnected rectilinear graph. For all interior faces f in the embedding of G, $\varphi(f) = (n-2)\times 180°$, and for the unique exterior face f_e, $\varphi(f_e) = (n+2)\times 180°$.*

Proof. Consider the embedding of G. The faces of the embedding are determined by G, and are simple polygons in the plane. By the definition of φ, for every interior we count the interior angles, and for the exterior face we count the exterior angles. The lemma follows. (Remember that if G is a simple cycle, the embedding has two faces). □

LEMMA 5.4. *Let G be an embeddable biconnected rectilinear graph, f_e the exterior face in its embedding and v a vertex on f_e. If $\varphi_{f_e}(v) = 180°$, then G can be embedded inside a polygon of shape* **U**, *as shown in* Fig. 5.4a. *If $\varphi_{f_e} = 270°$, then G can be embedded inside a polygon of shape* **W**, *as shown in* Fig. 5.4b.

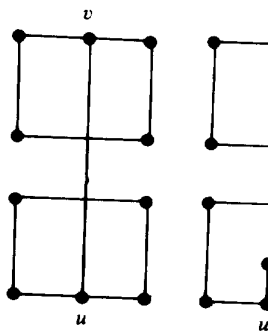
Proof. The proof is easy and left to the reader. □



(a)

6. Articulation ver
the embeddability of the
of the graph itself. Clear
vertices. In Fig. 3.2, we
each of which decompos

In those cases, the
articulation vertex. How
nonembeddable graphs,
components, so that the c
Note that an edge is a (



(a)

FIG

If v is an articulatio
connected subgraphs G_i o
meeting at v. Throughout
with rectilinear graphs wh

DEFINITION 6.1. Let
rectilinear graph G that s
interlace if the horizontal
(Fig. 3.2a). We also say tha
not have this property is s

LEMMA 6.1. *A rectili*
not embeddable.

Proof. Let B_1 and B_2
Since B_1 and B_2 are nontri

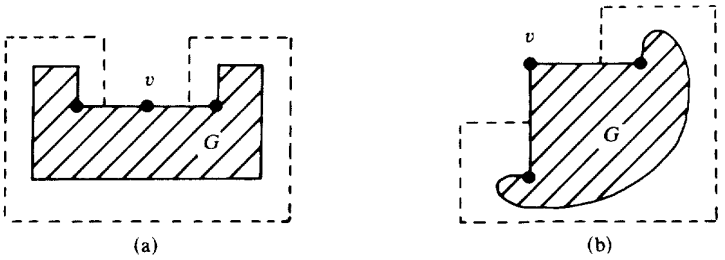


FIG. 5.4. *Shapes* U *and* W.

6. Articulation vertices. In this section we examine the conditions under which the embeddability of the biconnected components of the graph imply the embeddability of the graph itself. Clearly, this will depend on the way components meet at articulation vertices. In Fig. 3.2, we showed two examples of nonembeddable rectilinear graphs, each of which decomposes into two embeddable biconnected rectilinear graphs.

In those cases, the two biconnected components are not "compatible" at the articulation vertex. However, the situation need not be so local. Fig. 6.1 depicts two nonembeddable graphs, each of which decomposes into three embeddable biconnected components, so that the components meeting at each articulation vertex are compatible. Note that an edge is a (trivial) biconnected component.

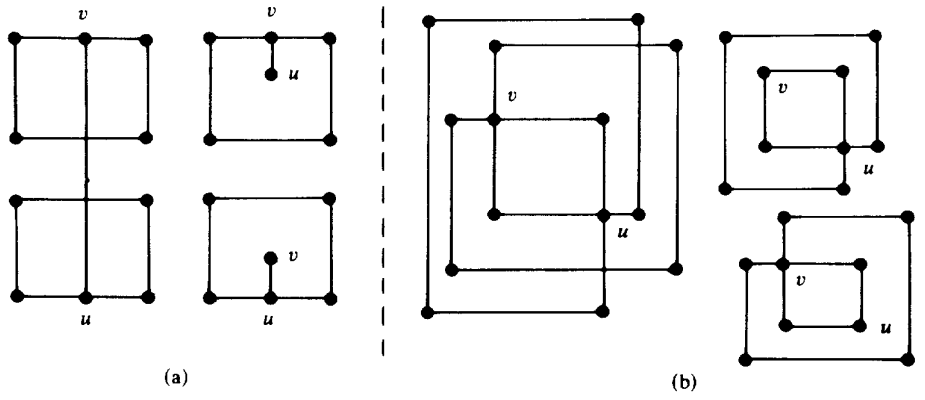


FIG. 6.1. *Decompositions of nonembeddable graphs.*

If v is an articulation vertex in a graph G, then its removal results in several connected subgraphs G_i of G. We will refer to the subgraphs $G_i + v$, as the subgraphs *meeting* at v. Throughout this section we will implicitly assume that we are dealing with rectilinear graphs whose biconnected components are embeddable.

DEFINITION 6.1. Let B_1 and B_2 be two nontrivial biconnected components of a rectilinear graph G that share an articulation vertex v. Then B_1 and B_2 are said to *interlace* if the horizontal edges at v belong to B_1 and the vertical edges belong to B_2 (Fig. 3.2a). We also say that v is an *interlace* vertex. Any articulation vertex that does not have this property is said to be *interlace-free.*

LEMMA 6.1. *A rectilinear graph G which has an interlace articulation vertex v is not embeddable.*

Proof. Let B_1 and B_2 be the two biconnected components sharing the vertex v. Since B_1 and B_2 are nontrivial, the horizontal edges at v lie on a cycle in B_1 and the

vertical edges lie on a cycle in B_2. It is impossible to draw G on the plane without these two cycles crossing. □

DEFINITION 6.2. Let B_1 and B_2 be two noninterlacing biconnected components of G that share an articulation vertex v, and assume B_1 is nontrivial. Then B_1 is said to *dominate* B_2 *at* v (or, B_2 is *inside* B_1) if either (i) v is not on the exterior face of B_1, or (ii) edges (v, u) and (v, w) at the vertex v are on the exterior face of B_1, and u, w are consecutive in that order in $L_G(v)$ (note that they are always consecutive in $L_{B_1}(v)$). If neither B_1 dominates B_2 nor B_2 dominates B_1, then B_1 and B_2 are said to be *outside* each other.

The intuition behind the above definition is that in the embedding, one biconnected component must lie wholly inside some face of the other if one edge of it does. This is due to the planarity criterion. Clearly, if biconnected components B_1 and B_2 that share an articulation vertex v dominate each other, the graph is not embeddable (this is the case in Fig. 3.2b).

Let B_1 and B_2 be two biconnected components of a graph G that share an articulation vertex v, such that B_1 dominates B_2. Let G' be the subgraph of G meeting at v that contains B_2. If G is embeddable then in any embedding of G, all of G' should lie inside one face of B_1. This suggests extending the relation "dominate" as follows:

DEFINITION 6.3. Let $\mathbf{B} = \{B_1, B_2, \cdots, B_m\}$ be the set of biconnected components of G. We say that B_i *dominates* B_j if there exists a biconnected component B_k and an articulation vertex v, such that (i) B_k and B_i share v, (ii) B_i dominates B_k at v, and (iii) B_j and B_k are both subgraphs of one of the connected subgraphs meeting at v.

Let us denote by $V(G)$ the vertex set of the graph G and by $E(G)$ the edge set.

LEMMA 6.2. *If in a rectilinear graph G, there exists some pair of biconnected components B_1 and B_2 that dominate each other, then G is not embeddable.*

Proof. If B_1 and B_2 share an articulation vertex v, then as mentioned earlier G is not embeddable. Suppose that B_1 and B_2 are disjoint. Since B_1 and B_2 dominate each other, there must be articulation vertices v_1, v_2, biconnected components B_1', B_2', and subgraphs G_1, G_2, such that for $i = 1, 2$, (i) B_i and B_i' share v_i, (ii) B_i dominates B_i' at v_i, and (iii) G_i is one of the subgraphs meeting at v_i and contains B_i'. Let us assume that G is embeddable. From (i) $v_2 \in V(G_1)$, (ii) G_1 lies wholly inside B_1 in the embedding, and (iii) $V(G_1) \cap V(B_1) = \{v_1\}$, we can conclude that v_2 must be properly inside a polygon defined by the face f_1 of B_1 containing v_1. Similarly v_1 should be properly inside the polygon defined by a face f_2 of B_2 containing v_2. Therefore some vertices of f_2 must lie outside f_1 and the two faces must intersect, and hence G is not embeddable. □

Given a rectilinear graph G, with a set of biconnected components $\mathbf{B}$ and a set of articulation vertices $\mathbf{A}$, we can construct a tree T of biconnected components such that

$$V(T) = \mathbf{A} \cup \mathbf{B}, \quad \text{and} \quad E(T) = \{(v, B) | v \in \mathbf{A}, B \in \mathbf{B}, v \in V(B)\}.$$

LEMMA 6.3. *Let G be a rectilinear graph with the set of biconnected components $\mathbf{B}$ and tree of biconnected components T. Let B be a leaf in the tree T which is adjacent to an articulation vertex v of degree 2 in T. If B dominates B' the other biconnected component adjacent to v in T, then B dominates every other biconnected component in $\mathbf{B}$.*

Proof. The only two subgraphs meeting at v are B and $G - B + v$ and the proof follows from Definition 6.3. □

If no two biconnected components dominate each other, then the relation "dominate" induces a partial order on $\mathbf{B}$. A nondominating element in this partial order is a biconnected component which does not dominate any biconnected component.

COROLLARY 6.1. *If*
there exists a nondomina
biconnected components.

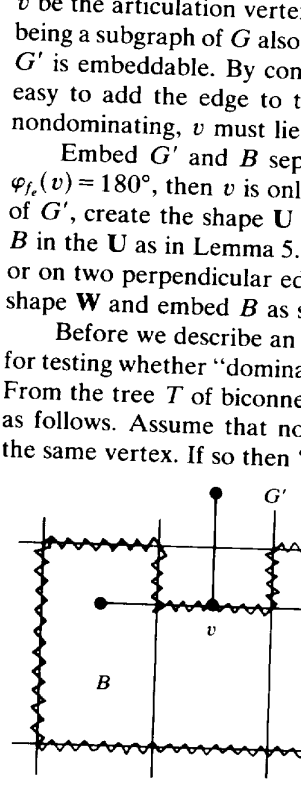
Proof. Any trivial b
dominating. If any vertex
to two leaves, then either
them is a trivial biconnec
then all leaves are adjac
leaves. If two of these
dominate each other whi
fact all of these leaves m

THEOREM 6.1. *Let G*
G is embeddable if and o
(i) *every biconnecte*
(ii) *every articulatio*
(iii) "*dominate*" *indi*

Proof. The necessary
The sufficient part is
induction is any biconnecte
n. Assume that the claim is
components. By Corollary
v be the articulation verte
being a subgraph of G also
G' is embeddable. By con
easy to add the edge to t
nondominating, v must lie

Embed G' and B sep
$\varphi_{f_e}(v) = 180°$, then v is onl
of G', create the shape **U**
B in the **U** as in Lemma 5.
or on two perpendicular ed
shape **W** and embed B as

Before we describe an
for testing whether "domina
From the tree T of biconne
as follows. Assume that no
the same vertex. If so then



(a)

FIG.

COROLLARY 6.1. *If for a rectlinear graph G "dominate" is a partial order, then there exists a nondominating biconnected component which is a leaf in the tree T of biconnected components.*

Proof. Any trivial biconnected component (which is just an edge) must be nondominating. If any vertex in T (corresponding to an articulation vertex in G) is adjacent to two leaves, then either the two leaves are nontrivial and not dominating, or one of them is a trivial biconnected component. If no vertex in T is adjacent to two leaves, then all leaves are adjacent to vertices of degree 2, and there are at least two such leaves. If two of these leaves are dominating, then by Lemma 6.3 the two leaves dominate each other which is a contradiction that "dominate" is a partial order. In fact all of these leaves must be nondominating. □

THEOREM 6.1. *Let G be a rectlinear graph and $\mathbf{B}$ its set of biconnected components. G is embeddable if and only if*

(i) *every biconnected component B in $\mathbf{B}$ is embeddable,*

(ii) *every articulation vertex in G is interlace-free, and*

(iii) *"dominate" induces a partial order on $\mathbf{B}$.*

Proof. The necessary part follows from Lemma 6.1 and Lemma 6.2.

The sufficient part is shown by induction on the number of vertices. The basis for induction is any biconnected rectilinear graph. Let G be not biconnected with $|V(G)| = n$. Assume that the claim is true for all smaller graphs. Look at the tree T of biconnected components. By Corollary 6.1, there exists a leaf B in T which is nondominating. Let v be the articulation vertex shared by B and $G' = G - B + v$, the rest of the graph. G' being a subgraph of G also satisfies the conditions of the claim. By induction hypothesis G' is embeddable. By condition (i), B is also embeddable. If B is a single edge it is easy to add the edge to the embedding of G'. Assume B is nontrivial. Since B is nondominating, v must lie on the exterior face f_e of B and $\varphi_{f_e}(v) \neq 90°$ (why?).

Embed G' and B separately and consider the vertex v in both embeddings. If $\varphi_{f_e}(v) = 180°$, then v is only one edge in G'. Add six new grid lines to the embedding of G', create the shape **U** as shown in Fig. 6.2a, magnify the embedding, and embed B in the **U** as in Lemma 5.4. If $\varphi_{f_e}(v) = 270°$, then v is either on just one edge in G', or on two perpendicular edges in G'. In both cases, add six new grid lines, create the shape **W** and embed B as shown in Fig. 6.2b. □

Before we describe an algorithm for testing embeddability, we need an algorithm for testing whether "dominate" is a partial order on the set of biconnected components. From the tree T of biconnected components, we construct $\bar{T}$ a partially directed tree as follows. Assume that no biconnected component dominates and is dominated at the same vertex. If so then "dominate" is not a partial order. Direct edge (v, B) from

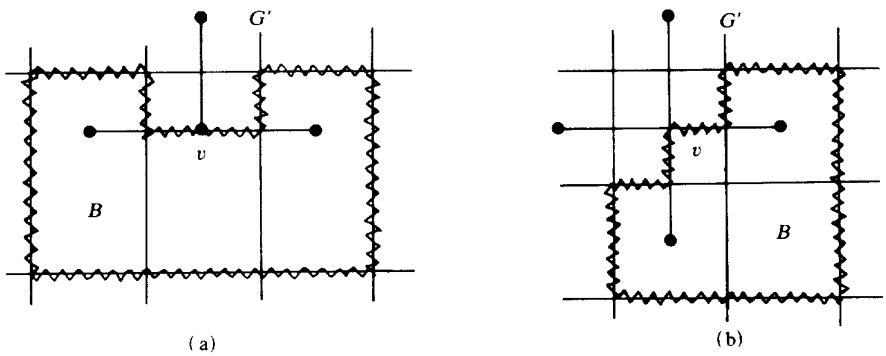


FIG. 6.2. *Adding B to the embedding of G'.*

B to v if B dominates at v. Direct edge (v, B) from v to B if B is dominated at v. Leave all the other edges undirected.

This partially directed tree $\bar{T}$ can be constructed in linear time as follows. Find the faces of each of the biconnected components using the algorithm *check-biconnected*. This takes $O(|V|)$ time. Check for dominations at each articulation vertex as described in Definition 6.2. There are at most 4 biconnected components at each articulation vertex and hence there are at most 12 (ordered) pairs to be tested for domination (in fact only 2 tests are necessary, how?). Construct $\bar{T}$ by directing the edges of T as described earlier. Note that articulation vertices and biconnected components can be found in $O(|V|)$ [1]. For each vertex x in $\bar{T}$, denote by $d_{\text{in}}(x)$, $d_{\text{out}}(x)$, and $d(x)$, the number of incoming arcs, the number of outgoing arcs, and the number of undirected edges of x respectively. The rest of the algorithm is given below.

```
Algorithm check-dominate-po (G);
begin
  construct T̄;
  for each vertex x in T̄ do
    if d_in(x) > 1 then
    begin
      write ("not a partial order");
      quit
    end;
  if search (T̄) then write ("yes, partial order")
  else write ("not a partial order")
end;

function search (T̄): boolean;
begin
  if T = Φ then search := true
  else begin
    if ∃ B ∈ B with d_out(B) = 0, d_in(B) + d(B) = 1 then
    begin
      Let v be the neighbor of B;
      if d_in(v) + d_out(v) + d(v) = 1 then search := search (T̄ − {B})
      else search := search (T̄ − {B, v})
    end else search := false
end.
```

The above algorithm can be easily shown to be correct using Definition 6.3 and Corollary 6.1. The boolean function *search* can be implemented nonrecursively to run in linear time by maintaining a queue of the leaves of $\bar{T}$.

Given the biconnected components and articulation vertices, checking that the articulation vertices are interlace-free can be done in $O(|V|)$ time. Let *check-interlace-free* be a procedure that checks a given articulation vertex for interlace-freedom. We end this section with a $O(|V|)$ algorithm for testing embeddability of rectilinear graphs.

```
Algorithm check-rectilinear (G);
begin
  Decompose G into its biconnected components;
  for each biconnected component B do check-biconnected (B);
  for each articulation vertex v do check-interlace-free (v);
  check-dominate-po (G)
end.
```


7. An embedding
testing embeddability.
gives an embedding. Ho
The reasoning is as fo
could be $O(|V|)$ long.
transform it to the path
once. Thus for each pa
paths and hence the co
to $O(|V|)^2)$, we have t
than the path P. In this
the $O(|V|^2)$ complexity
such paths, describe the

LEMMA 7.1. *Let G*
Then any embedding of

Proof. The dual G^d
G^d is a simple graph. I
biconnected and hence o
than or equal to 5. □

LEMMA 7.2. *Given*
a cycle, there is a simple
2, (ii) *the end vertices of*
planar embedding, and (i

Proof. As in the proo
with property (i) and (ii)
at most 5. The longest of
conditions (iii) and (iv).

To get an embedding
embeddable and then app

```
Algorithm embed-rect
begin
  for each biconnecte
  join-the-embeddings
end.

Algorithm embed-bico
begin
  get-long-path (P, P1,
  embed-rectilinear (B
  find-path-in-embeddi
  apply-operations-and
end.
```

Procedure *get-long-pat*
the conditions of Lemma 7.2,
such a path exists.

Procedure *find-path-in-*
such that P' starts and ends
and P are both complements
face, P' can be obtained by s
P_1 in the embedding of $B - P$
result in P' being a complemer

7. An embedding algorithm. In the previous section we gave an algorithm for testing embeddability. This algorithm can be easily modified into an algorithm which gives an embedding. However, the complexity of this naive algorithm would be $O(|V|^3)$. The reasoning is as follows. The path P' that we find in the proof of Theorem 5.1 could be $O(|V|)$ long. For each topological operation that we apply on this path to transform it to the path P, we update the coordinates of the vertices in the embedding once. Thus for each path added we require $O(|V|^2)$ time. There can be $O(|V|)$ such paths and hence the complexity of the algorithm is $O(|V|^3)$. To reduce the complexity to $O(|V|)^2)$, we have to make sure that the path P' is never longer (asymptotically) than the path P. In this case the sum of the lengths of all such paths P' is $O(|V|)$, and the $O(|V|^2)$ complexity follows. In the following, we show how we can always find such paths, describe the algorithm, and analyze its complexity.

LEMMA 7.1. *Let G be a planar biconnected multigraph with minimum degree three. Then any embedding of G has an interior face of size at most five.*

Proof. The dual G^d of G is also a planar graph. Since G has minimum degree 3, G^d is a simple graph. Hence G^d has at least two vertices of degree $\leqq 5$ [2]. G is biconnected and hence one of the vertices must correspond to a face whose size is less than or equal to 5. □

LEMMA 7.2. *Given an embedding of a planar biconnected graph G, which is not a cycle, there is a simple path P, such that* (i) *the interior vertices of P all have degree* 2, (ii) *the end vertices of P have degree* $\geqq 2$, (iii) *P appears in an interior face f in the planar embedding, and* (iv) $5 \cdot |P| \geqq |f|$.

Proof. As in the proof of Lemma 5.2, transform G to G' by replacing all paths with property (i) and (ii) by edges. By Lemma 7.1, G' has an interior face f of size at most 5. The longest of all the paths in G corresponding to the edges of f will satisfy conditions (iii) and (iv). □

To get an embedding of a given rectilinear graph, we first test if the graph is embeddable and then apply the following algorithm.

Algorithm *embed-rectilinear* (G);
begin
  for each biconnected component B **do** *embed-biconnected* (B);
  join-the-embeddings;
end.

Algorithm *embed-biconnected* (B);
begin
  get-long-path (P, P_1, σ);
  embed-rectilinear $(B-P)$;
  find-path-in-embedding (P', P_1, σ);
  apply-operations-and-transform (P', P)
end.

Procedure *get-long-path* returns paths P, P_1, and square σ, such that P satisfies the conditions of Lemma 7.2, and the interior face $f = PP_1$ simplifies to σ. By Lemma 7.2 such a path exists.

Procedure *find-path-in-embedding* traces a path P' in the embedding of $B-P$, such that P' starts and ends in the same directions as P, and $P'P_1$ simplifies to σ. P' and P are both complements of P_1 with respect to the square σ. Since PP_1 is an interior face, P' can be obtained by starting in the required direction, then following the path P_1 in the embedding of $B-P$, and ending in the required direction (Fig. 7.1). This will result in P' being a complement of P_1 with respect to σ. We have $|P'| = O(|P_1|) = O(|P|)$.

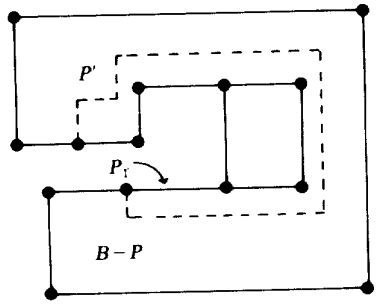


FIG. 7.1. *Finding the path P' in the embedding of B − P.*

Procedure *apply-operations-and-transform* applies a sequence of the four topological operations to P' in the embedding of $B - P + P'$ and transforms it to P thus resulting in a embedding of B. This is done by first simplifying the path P' and then expanding the simplified path to get the path P (Fig. 7.2). The number of operations applied will be $O(|P| + |P'|) = O(|P|)$.

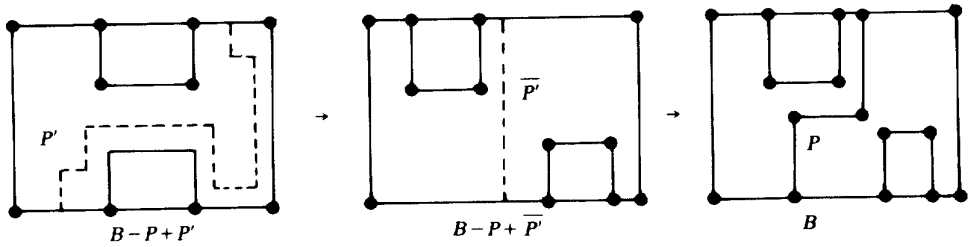


FIG. 7.2. *Path addition, simplification and expansion.*

Procedure *join-embeddings* takes the embeddings of the biconnected components and puts them together to get an embedding for G. This is done essentially following the proof of Theorem 6.1. Find a nondominating component B. Recursively embed $G' = G - B$. Join the embeddings of B and G' using the shapes **U** or **W** as shown in Fig. 6.2.

The algorithm can be shown to be correct using the material developed in the previous three sections. We now analyze the complexity of each step in the algorithm and show that the total complexity is $O(|V|^2)$.

Procedure *join-embeddings* updates each coordinate at most once per recursive call. The total number of calls is bounded by the number of biconnected components. Hence this procedure takes $O(|V|^2)$ time.

Procedure *get-long-path* can be implemented to run in $O(|V|)$ time each time it is called. Remember that we can get the faces of a biconnected graph from the testing algorithm, and searching all faces to get the required face takes linear time. Procedure *find-path-in-embedding* takes $O(|P_1|) = O(|V|)$ time. These two procedures will be invoked at most $O(|V|)$ time. Hence total time spent in these calls is $O(|V|^2)$.

Procedure *apply-operations-and-transform* applies a sequence of $O(|P|)$ operations. Each edge in G will appear in only one such path P. Hence the sum of the lengths of all such paths P is $O(|V|)$. Each operation updates at most $O(|V|)$ coordinates. Therefore the time spent in calls to this procedure is $O(|V|^2)$.

8. Consistent rectilinear graphs. Certain rectilinear graphs cannot be drawn on the grid even if we relax the planarity criterion. We say that a rectilinear graph $G(V, E, \lambda)$ is *consistent* if it can be drawn on the grid satisfying the ordering relation λ. In other words, G is consistent if the set of equality and inequality constraints generated in part 1 of Definition 2.2 is consistent.

The equality constraints define an equivalence relation on the set of coordinates of the vertices of G. Let us denote by $e(x)$ the equivalence class containing the coordinate x. Denote by I_x and I_y the sets of x-coordinate and y-coordinate inequality constraints respectively. Construct two directed graphs $G_x(V_x, E_x)$ and $G_y(V_y, E_y)$ as follows:

$$V_x = \{e(x) | x = x(a), a \in V\} \quad \text{and} \quad E_x = \{(x_1, x_2) | x_1 > x_2 \in I_x\}.$$

V_y and E_y are similarly defined.

It can be easily shown that G is consistent if and only if the two directed graphs G_x and G_y are both acyclic. A solution to the coordinates, which satisfies the constraints will correspond to a (possibly) nonplanar embedding of G on the grid. This can be obtained by performing the topological sort operation [5] on the two acyclic digraphs. In fact this will yield a solution that minimizes the area of the rectangle bounding the embedding.

In a nonplanar embedding of a consistent rectilinear graph on the grid, all crossings are between horizontal edges and vertical edges. The vertical edges can be assigned one layer, and the horizontal edges can be assigned a second layer. In other words the "thickness" [2] of a consistent rectilinear graph is less than or equal to two.

9. Extensions, open problems, and conclusions.

1. It can be easily shown that the area of the embedding given by the algorithm in this paper can be made $O(|V|^2)$ without extra time penalty. There are graphs that require this much area. To minimize the area is NP-complete if the input graph is allowed to be disconnected. The minimization problem is open for connected rectilinear graphs.

2. The embedding problem of appropriately defined graphs for other grids (triangular, hexagonal, etc.), seems to be interesting in light of certain systolic layouts for VLSI [4] that use them. It originally seemed to us that the ideas of this paper will carry through without much change to other grids. They do not. "Triangular" graphs, for example, may have triangles which must be equilateral in any embedding. This rigidity (which does not appear in the rectilinear case), makes some of our results false for these graphs.

3. If we allow two layers for the embedding (each of which must be planar), then assigning horizontal and vertical edges to different layers easily solves the problem. However, in reality the user decides which wire will be on which layer. The results in this paper give only a necessary condition for the embeddability of such a multilayered rectilinear graph. Under what conditions can we obtain compatible embeddings on the different layers (i.e. embeddings that have corresponding vertices at the same grid points)?

4. An interesting class of graphs that contains all rectilinear graphs is the class of graphs in which the edges incident on each vertex are cyclically ordered (now there is no degree or direction constraints). The corresponding problem is whether a graph in this class can be laid on the plane consistent with the cyclic orderings of the edges at each vertex, so that no edges cross. This problem can be solved in linear time [11].

We conclude by observing that even linear time and space algorithms may not be considered efficient for VLSI applications, due to the huge size of the graphs involved.

However, if the layout is given by a "good" hierarchical description, then both time and space complexity of our algorithms can be reduced considerably. ALI allows hierarchical description of layouts through its *cell* mechanism [8], and our algorithms will be implemented in ALI.

10. Acknowledgments. This problem was originally raised by Professors Bob Sedgewick and Dick Lipton. We wish to thank Professors Dick Lipton and Jacobo Valdes for several useful discussions. We are grateful to Professor George Lueker for pointing out an omission in an earlier version of this paper. Our thanks are also due to Edna Wigderson, Vijaya Ramachandran and the referee for their comments.

REFERENCES

[1] A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[2] M. BEHZAD, G. CHARTRAND AND L. LESNIAK-FOSTER, *Graphs and Digraphs*, Wadsworth International Group, London, 1981.

[3] J. E. HOPCROFT AND R. E. TARJAN, *Efficient planarity testing*, J. Assoc. Comput. Mach., 21 (1974), pp. 549-568.

[4] H. T. KUNG AND C. E. LEISERSON, *Systolic arrays (for* VLSI) in Sparse Matrix Proceedings, I. S. Duff and G. W. Stewart eds., Society for Industrial and Applied Mathematics, Philadelphia, 1979.

[5] D. E. KNUTH, *The Art of Computer Programming, Vol.* 1, *Fundamental Algorithms*, Addison-Wesley, Reading, MA, 1971.

[6] C. E. LEISERSON, *Area efficient graph embeddings (for* VLSI), Proc. 21st Symposium on the Foundations of Computer Science, October 1980.

[7] R. J. LIPTON, S. C. NORTH, R. SEDGEWICK, J. VALDES AND G. VIJAYAN, VLSI *layout as programming*, ACM Trans. Programming Languages and Systems, 5 (1983), pp. 405–421.

[8] ———, ALI: *a procedural language to describe* VLSI *layouts*, Proc. of the 19th Design Automation Conference, Las Vegas, June 1982.

[9] R. SETHI, *Testing for the Church-Rosser property*, J. Assoc Comput. Mach., 21 (1974), pp. 671–679; *errata*, 22 (1975), p. 424.

[10] G. VIJAYAN, *Completeness of* VLSI *layouts*, VLSI Memo #1, Dept. of Electrical Engineering and Computer Science, Princeton Univ., Princeton, NJ, September 1982.

[11] G. VIJAYAN AND A. WIGDERSON, *Planarity of edge ordered graphs*, Technical Report #307, Dept. Electrical Engineering and Computer Science, Princeton Univ., Princeton, NJ, December 1982.

[12] A. WIGDERSON, *The complexity of the Hamiltonian circuit problem for maximal planar graphs*, Technical Report #298, Dept. Electrical Engineering and Computer Science, Princeton Univ., Princeton, NJ, February 1982.

SIAM J. COMPUT.
Vol. 14, No. 2, May 1985

AXIOMS FO

Abstract. In the standar
Its Syntax and Semantics, N
Introduction to Combinatory L
defined as a primitive operati
that the axioms for the theory
needed at all, as it is reduced
simplicity of the lambda-notat
in formulating the axiom syste
that replaces every (free or bo
program written in PL/I that

Key words. lambda-calcul

1. Introduction. Int
years especially among c
of programming languag
lies in its ability to hand

Substitution has alw
programming languages,
how to define the subst
complex if we have to dea
may be bound, as is the
of the book *Combinatory*
way: "The extent of the
most formulations of the
predicate calculus which
and there is little doubt t
Church {IML}, p. 57, was
form of combinatory logi

Various attempts have
most radical approach is
by Curry, where bound
combinators, to describe th
we have to sacrifice the in
of eliminating all variable
Unfortunately, all attemp
calculus or combinatory
substitution can be elimina
properties of identity in fo

In the present paper
Namely, we show that the
lated in such a way that s
essential use of the propert
same time, in contrast with
of the lambda-notation. Ou

* Received by the editors No
† Computer Science Departm