A PROVABLY GOOD LINEAR ALGORITHM FOR EMBEDDING GRAPHS IN THE RECTILINEAR GRID

Roberto Tamassia
Ioannis G. Tollis
In this paper we consider planar embeddings of n-node planar graphs in the rectilinear grid, where vertices are grid points and edges are nonintersection grid paths. We present a new embedding algorithm that runs in linear time. The total number of bends in the embeddings constructed by our algorithm is very small. Furthermore, the embeddings occupy $O(n)$ area, which is the best possible in the worst case. Our results are important in the design of VLSI chips. Other applications can be found in the areas of communication by light or microwave, transportation in space, and aesthetic layout of diagrams.
A PROVABLY GOOD LINEAR ALGORITHM FOR EMBEDDING GRAPHS IN THE RECTILINEAR GRID

by

Roberto Tamassia *
Coordinated Science Laboratory and
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

Ioannis G. Tollis **
Coordinated Science Laboratory and
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

ABSTRACT

In this paper we consider planar embeddings of $n$ -node planar graphs in the rectilinear grid, where vertices are grid points and edges are nonintersecting grid paths. We present a new embedding algorithm that runs in linear time. The total number of bends in the embeddings constructed by our algorithm is very small. Furthermore, the embeddings occupy $O(n^2)$ area, which is the best possible in the worst case. Our results are important in the design of VLSI chips. Other applications can be found in the areas of communication by light or microwave, transportation in space, and aesthetic layout of diagrams.

---

* On leave from: Dipartimento di Informatica e Sistemistica - Universita' di Roma, "La Sapienza" - Via Buonarroti 12 - 00185 Roma, Italy. This author was supported in part by Progetto Finalizzato Trasporti of the Italian National Research Council.

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

The problem of embedding a graph in the rectilinear grid has been extensively studied in the literature. Here, we consider planar embeddings, where vertices are grid points and edges are nonintersecting grid paths. Graph embeddings are of particular importance in VLSI circuit layout problems [9, 17, 18]. Three important measures of quality for graph embeddings are [9]:

1. The area of the smallest rectangle enclosing the embedding.
2. The number of bends along the edges.
3. The total edge length and the length of the longest edge.

Clearly, minimizing the above cost measures is very important in the design of VLSI chips [1, 5, 9, 13]. To avoid complications, due to the fact that every node in the grid has degree four, we assume that every node of a graph has degree less than or equal to four.

An embedding with crossings means that more than one conducting layers need to be used for edges that cross in order to avoid contact between two different wires. Although the recent advances in fabrication technology allow the use of two or more layers in VLSI, several reasons make crossing-free layouts desirable [4]. Furthermore, a connection between different layers (via) has a significant cost in area, reliability, and performance.

The node-cost measure for these embeddings is the total number of bends that are present along the edges. Figure 1 illustrates an example of an embedding with node-cost equal to 4. Minimizing the node-cost has applications in VLSI, as discussed in the previous paragraph, communication by light or microwave, transportation in space [11], and aesthetic layout of diagrams [2].

This problem was first considered by Storer [12, 13], who conjectured it to be NP-hard and presented three heuristics proving worst-case bounds on their performance. Namely, he proved that the total number of bends of the embeddings produced by his algorithms is at most $2n$ if $G$ is biconnected and $2.4n + 4$ otherwise. Although Storer was not concerned with the exact
implementation and time complexity of his heuristics, he pointed out that they can be implemented in polynomial time. He did not consider the area occupied by the embeddings produced by his algorithms. Later, Tamassia [14] disproved Storer’s conjecture giving an exact algorithm that computes an embedding of a graph $G$ with $n$ vertices in $O(n^2)$ time. His algorithm uses network flows techniques and produces an embedding of $G$ with the minimum number of bends.

Recently, Aggarwal, Klawe, Lichtenstein, Linial, and Wigderson [1] presented an algorithm which, given an embedding of a planar graph in the plane, constructs a topologically equivalent $O(n^2)$ area (rake) grid embedding with the property that every edge makes at most six bends. However, they do not analyze the running time of their algorithm.

In this paper, we present a new embedding algorithm that runs in linear time. This is the first linear time algorithm, to the best of our knowledge, that achieves the following desirable properties. All the grid embeddings produced by our algorithm occupy area $O(n^2)$, which is the best possible in the worst case [19], and have a number of bends which is comparable to that achieved by Storer. Namely, the total number of bends is at most $2n + 4$ if $G$ is biconnected, and $2.5n + 2$ otherwise. This constitutes a significant improvement upon the results of [1] as far as the time
complexity and the node-cost measure are concerned, and upon the results of [13] as far as the time complexity and the area measures are concerned. The algorithm is based on the construction of a \textit{w-visibility representation} [15] for the graph to be embedded, which consists of representing vertices by horizontal segments and edges by vertical segments connecting the horizontal segments associated to their endpoint vertices.

Section 2 contains formal definitions and preliminary results. In section 3 we describe an algorithm that constructs the \textit{w-visibility representation}. In Section 4 we describe three transformations that reduce the node-cost of an embedding. Section 5 presents the algorithm and analyzes its performance and complexity. Conclusions and open problems are discussed in Section 6.

2. PRELIMINARIES

The basic definitions on graphs and planarity can be found in many textbooks, see for example [3, 7]. Here we define various representations of graphs in the plane in order to avoid confusion.

A \textit{plane graph} is a graph such that:

1. vertices are distinct points of the plane;
2. edges are simple curves connecting their endpoint vertices; and
3. edges do not cross each other, except possibly at common endpoints.

A graph is \textit{planar} if it is isomorphic to some plane graph.

The \textit{planar representation} of a plane graph \( G = (V, E) \) is the set of its adjacency lists \( P = \{P(v), \ v \in V\} \), where each list \( P(v) \) is circularly ordered according to the clockwise sequence of edges around vertex \( v \). Let \( e \) be an element of \( P(v) \). It consists of three fields \( \alpha(e), \sigma(e), \) and \( \mu(e) \) where: \( \alpha(e) \) is the adjacent vertex, \( \sigma(e) \) is the next element in the list \( P(v) \), and \( \mu(e) \) is the symmetric adjacency element in the list \( P(\alpha(e)) \). The planar representation describes the \textit{topology} of the plane graph \( G \).
An orthogonal graph is a plane graph whose edges are alternating sequences of horizontal and vertical segments. The orthogonal representation of an orthogonal graph $G = (V, E)$ describes the shape of $G$ without considering segment lengths. It consists of a set of "enriched" adjacency lists $H = \{ H(v), v \in V \}$, where each list $H(v)$ is obtained from the list $P(v)$ by adding two new fields $\theta(e)$ and $\rho(e)$ to all elements $e$, with the following meaning:

1. $\theta(e)$ is the angle that edge $(v, \alpha(e))$ forms with edge $(v, \alpha(\sigma(e)))$ at vertex $v$, measured in $\pi/2$ units;
2. $\rho(e)$ is the number of $\pi/2$ angles that appear on the left side of edge $(v, \alpha(e))$ when going from $v$ to $\alpha(e)$.

The node-cost $N(G)$ of an orthogonal graph $G = (V, E)$ is the total number of bends formed by its edges. Namely, if $H$ is the orthogonal representation of $G$, the node cost is:

$$N(G) = \sum_{v \in V} \sum_{e \in H(v)} \rho(e).$$

A grid graph is an orthogonal graph whose segments have all integer length. A grid embedding for a graph $G$ is a grid graph isomorphic to $G$. Planar embeddings and orthogonal embeddings are defined similarly. It is easy to see that a graph admits an orthogonal embedding if and only if it is planar and has degree less than or equal to four.

**Lemma 1** Let $H$ be the orthogonal representation of an orthogonal graph $G$ with $n$ vertices. Then a grid embedding of $G$ with orthogonal representation $H$ can be constructed from $H$ in time $O(n + N(G))$.

**Proof:** See [14]. □

3. W-VISIBILITY REPRESENTATION

A w-visibility representation [15] for a graph $G = (V, E)$ is a mapping of vertices of $G$ into nonoverlapping horizontal segments (called vertex segments) and of edges of $G$ into vertical
segments such that, for each edge \((u,v) \in E\), the associated edge-segment has its endpoints on the vertex-segments corresponding to \(u\) and \(v\), and it does not cross any other vertex segment.

We present now an algorithm for constructing \(w\)-visibility representations. First, we give some definitions on PERT-digraphs and \(st\)-numberings. Next, we describe a linear time algorithm for constructing a \(w\)-visibility representation of a planar 2-connected graph \(G = (V,E)\). For the sake of simplicity, we will use the same notation for the vertex-segments of the visibility representations and their corresponding vertices in the graph. The same will be done for the edge-segments and their corresponding edges.

A PERT-digraph \(D = (V,A)\) is an acyclic digraph with exactly one source, \(s\) and one sink, \(t\). We usually associate a positive length with each arc of \(D\). A well-known problem on PERT-digraphs is the following: For each vertex \(v\) of \(D\), find the length of the longest path from \(s\) to \(v\). This quantity will be denoted by \(\lambda(v)\). The critical path method solves this problem in \(O(|A|)\) time. [7].

An \(st\)-numbering for a graph \(G = (V,E)\), where \(s\) and \(t\) are two distinct vertices of \(G\), is a one-to-one mapping \(\xi: V \to \{1,2, \ldots ,|V|\}\) such that: \(\xi(s) = 1\), \(\xi(t) = |V|\), and each vertex \(v \neq s,t\) has two adjacent vertices \(u, w\) for which \(\xi(u) < \xi(v) < \xi(w)\). Given an \(st\)-numbering \(\xi\) for a graph \(G = (V,E)\), we construct a digraph \(D = (V,A)\) by orienting every edge from the lowest numbered vertex to the highest one. Namely, \([u,v] \in A\) if and only if \((u,v) \in E\) and \(\xi(u) < \xi(v)\). The digraph \(D\), which is induced by \(\xi\), is clearly acyclic and has exactly one source, \(s\), and one sink, \(t\), i.e. it is a PERT-digraph. Lempel, Even, and Cederbaum [10] showed that, for every 2-connected graph and every edge \((s,t)\), there exists an \(st\)-numbering. A linear time algorithm for finding it has been presented by Even and Tarjan [6].

**Algorithm W-VISIBILITY**

**Input:** A planar 2-connected graph \(G = (V,E)\).
Output: A w-visibility representation for G such that each vertex- and edge-segment has endpoints with integer coordinates.

(1) Select an edge \((s,t) \in E\) and compute an \(st\)-numbering for \(G\). Let \(D\) be the directed graph induced by the \(st\)-numbering.

(2) Find a planar representation \(\hat{D}\) of \(D\) such that the arc \([s,t]\) is on the external face and the rest of \(D\) lies on the right side of \([s,t]\). Use \(\hat{D}\) to construct a new digraph \(D'\) as follows: Vertices of \(D'\) are the faces of \(\hat{D}\). There is an arc \([f,g]\) in \(D'\) if face \(f\) shares an arc \(a = [v,w]\), distinct from \([s,t]\), with face \(g\) and \(a\) is positively oriented with respect to \(f\), i.e., face \(f\) is on the left side of \(a\), when \(a\) is traversed from the tail to the head. Note that \(D'\) is a 2-connected planar PFRT-digraph, with source, \(s'\), (the internal face containing arc \([s,t]\)) and sink, \(t'\), (the external face).

(3) Apply the critical path method to \(D'\) with all arc-lengths equal to 2. This gives the function \(\lambda(f)\) for each vertex \(f\) of \(D'\).

(4) Construct the w-visibility representation as follows:

(4.1) Use the \(st\)-numbering computed in step 1 to assign y-coordinates to horizontal vertex-segments.

(4.2) Set the x-coordinate of arc \([s,t]\) equal to -1.

(4.3) For any other arc \(a\) of \(\hat{D}\), set the x-coordinate of the corresponding vertical edge-segment equal to an integer \(j\), with \(\lambda(f) < j < \lambda(g)\), where \(f\) and \(g\) are the faces of \(\hat{D}\) sharing \(a\) in their contour.

(4.4) Set the y-coordinates of the endpoints of each edge-segment equal to the ones of the connected vertex-segments.

(4.5) Set the x-coordinates of the left and right endpoint of each vertex-segment equal to the minimum and maximum x-coordinates of their incident arcs, respectively. If a vertex-segment \(v\) is incident to exactly two edge-segments with the same x-coordinate, \(x_v\), then
set the x-coordinates of the endpoints of \( v \) to \( x_i - 1 \) and \( x_r \), respectively. □

An example of the construction performed by the algorithm \textit{W-VISIBILITY} is given in fig. 2. Figure 2.a shows a planar embedding \( \hat{D} \) along with the corresponding \( D^* \). Vertices of \( \hat{D} \) and \( D^* \) are represented by white and black circles, respectively. The white vertices are numbered according to the \( st \)-numbering. For each black vertex \( f_i \), the value of \( \lambda(f_i) \) is shown in parentheses. Figure 2.b illustrates the \( w \)-visibility representation produced by the algorithm.

\textbf{Lemma 2} Let \( G \) be a planar 2-connected graph. Then, in the \( w \)-visibility representation produced by algorithm \textit{W-VISIBILITY} every vertex-segment, except the topmost and bottommost, has at least an edge-segment incident from above and an edge-segment incident from below.

\textbf{Proof:} See [15]. □

The above algorithm can be extended to work for a 1-connected graph without increasing the time complexity. Simply construct the \( w \)-visibility representation for each block and then combine
them appropriately. For further details see [15] or [16].

Lemma 3 A graph admits a w-visibility representation if and only if it is planar. Furthermore, there is an algorithm that constructs a w-visibility representation for an n-vertex planar graph in time $O(n)$.

Proof: See [15]. □

4. BEND-STRETCHING TRANSFORMATIONS

In this section we describe three transformations that, when applied to an orthogonal representation $H$, produce a new orthogonal representation $H'$ with the same underlying planar representation, but of lower node-cost. We introduce these transformations by means of examples. Angles of $\pi/2$ and $3\pi/2$ will be called convex and concave, respectively.

Let $G$ be the orthogonal embedding shown in fig. 3.a. and $H$ be its orthogonal representation. Edge $(u,v)$ has convex angles both on the left and right side, namely, two angles on the left, and one on the right. Then, as shown in fig. 3.b, we can modify $G$ so that $(u,v)$ has convex angles only on one side, namely, one angle on the left. Note that the new orthogonal representation $H'$ is different from $H$ only in the two elements corresponding to edge $(u,v)$. The transformation from $H$ to $H'$ is said to be of type TI.

Figure 3
Consider the orthogonal embedding shown in fig. 4.a. All edges incident to vertex $u$ have a convex angle on the right side. These angles can be removed easily as shown in fig. 4.b. The corresponding transformation of orthogonal representations is said to be of type $T2$.

Figure 5 illustrates a transformation of type $T3$. In the orthogonal graph of fig. 5.a edge $(u, w)$ follows edge $(u, v)$ in the clockwise sequence around vertex $v$, and forms with it an angle of $\pi$ in $u$. Furthermore, edge $(u, w)$ has a convex angle on the left side. This angle can be removed as shown in fig. 5.b.

The “bend-stretching” transformations $T1$, $T2$, and $T3$ are formally described in the following. For each transformation, we give the conditions of applicability, and the operations to be performed. $H$ denotes the orthogonal representation to which the transformations are applied; $e$ and $e'$ are elements of the lists of $H$. 

---

**Figure 4**

---

**Figure 5**
Transformation $T1$

Condition: $e' = \mu(e), \rho(e) \geq 1, \rho(e') \geq 1.$

Procedure: $x := \rho(e); y := \rho(e');$

$\rho(e) := \max \{0, x - y\};$

$\rho(e') := \max \{0, y - x\}.$

Transformation $T2$

Case 1

Condition: $\rho(e) > 0$ for all $e \in H(v)$.

Procedure: $m := \min_{e \in H(v)} \rho(e);$

for all $e \in H(v)$ do

$\rho(e) := \rho(e) - m.$

Case 2

Condition: $\rho(\mu(e)) > 0$ for all $e \in H(v)$.

Procedure: $m := \min_{e \in H(v)} \rho(\mu(e));$

for all $e \in H(v)$ do

$\rho(\mu(e)) := \rho(\mu(e)) - m.$

Transformation $T3$

Case 1

Condition: $e' = \sigma(e), \theta(e) \geq 2, \rho(e') \geq 1.$

Procedure: $m := \min \{\theta(e) - 1, \rho(e')\};$

$\theta(e) := \theta(e) - m;$

$\theta(e') := \theta(e') + m;$

$\rho(e') := \rho(e') - m.$
Case 2

**Condition:** \( e' = \sigma(e), e = \sigma(e''), \theta(e) \geq 2, \rho(\mu(e)) \geq 1. \)

**Procedure:**

\[
\begin{align*}
      m &= \min \{ \theta(e) - 1, \rho(\mu(e)) \}; \\
      \theta(e) &= \theta(e) - m; \\
      \theta(e'') &= \theta(e'') + m; \\
      \rho(\mu(e)) &= \rho(\mu(e)) - m.
\end{align*}
\]

Any transformation of type \( T1 \) reduces the node-cost at least by 2, while transformations of type \( T2 \) and \( T3 \) reduce it at least by 1.

**5. THE GRID EMBEDDING ALGORITHM**

In this section, we present the algorithm **GRID-EMBEDDING** that produces grid embeddings with low node-cost in linear time. The basic strategy of **GRID-EMBEDDING** consists of finding a \( w \)-visibility representation for the graph and then transforming it into a grid embedding.

**Algorithm ** **GRID-EMBEDDING**

**Input:** An \( n \)-vertex planar graph \( G = (V, E) \) of degree less than or equal to 4.

**Output:** A grid embedding of \( G \) with "low" node-cost.

1. Construct a \( w \)-visibility representation \( \Gamma \) for \( G \).

2. Transform \( \Gamma \) into an orthogonal embedding \( G_h \) of \( G \) by substituting each vertex segment of \( \Gamma \) with one of the structures shown in fig. 6. Let \( H \) be the orthogonal representation of \( G_h \).

3. Simplify the orthogonal representation \( H \) by means of the bend-stretching transformations as follows:

   3.1 First, perform transformation \( T1 \) on every edge to which it is applicable.
(3.2) Then, perform transformation $T_2$ on every vertex to which it is applicable.

(3.3) Finally, for each vertex $v$ of degree less than or equal to 3, perform transformation $T_3$ until it is no longer applicable. Let $H'$ be the orthogonal representation so obtained.
Figure 7 shows a simple example of a run of the GRID-EMBEDDING algorithm. The given graph is shown in part (a). Part (b) shows the w-visibility representation constructed by the algorithm W-VISIBILITY. Parts (c) and (d) show how the w-visibility representation is transformed into an orthogonal representation and into a grid embedding, respectively.

Lemma 4 The total number of transformations of type T3 performed by the algorithm GRID-EMBEDDING is at most 2n.

Proof: After the execution of the transformations of type T1, each edge has convex angles only on one side. Furthermore, after the execution of the transformations of type T2, the edges incident to any vertex do not all have convex angles on the same side. Let v be a vertex of degree 3. Assume, without loss of generality, that one edge incident to v, say (v, w), has convex angles on the left side, while the other two edges have convex angles, if any, on the right side. The edges incident to v form exactly one angle of π and two angles of π/2. We have three cases for the location of the angle of π, as shown in figs. 8.a, 8.b, and 8.c. We have that at most two transformations of type T3 can be performed in the first case, one in the second, and none in the third. With similar arguments it can be shown that at most two transformations of type T3 are performed on any vertex of degree one or two. □

We give now the time complexity analysis for algorithm GRID-EMBEDDING. From lemma 3, step 1 can be performed in $O(n)$ time. Clearly, step 2 takes $O(n)$ time too. At most 2n transformations of type T1, n of type T2, and, by lemma 4, 2n of type T3 are performed in step 3. Thus, step 3 also has complexity $O(n)$. From lemma 1, step 4 takes $O(n + N(G'))$. Finally, the execution of step 3 does not increase the node-cost, so that $N(G') \leq N(G_n)$. Therefore, since the transformation of step 2 is such that $N(G_n) = O(n)$, we have:
Theorem 1 The overall complexity of algorithm GRID-EMBEDDING is $O(n)$. The basic results on the performance of our algorithm are presented in the next three theorems.

Theorem 2 Let $G = (V, E)$ be an $n$-vertex 2-connected graph of degree less than or equal to four, and $n_i$ be the number of vertices with degree $i$. Then the algorithm GRID-EMBEDDING produces a grid embedding $G'$ for $G$ with node-cost $N(G') \leq n_3 + 2n_4 + 4 \leq 2n + 4$.

Proof: The orthogonal embedding $G_h$ produced by step 2 has node-cost $N(G_h) \leq n_2 + 2n_3 + 4n_4$, as can be seen from figure 6. Step 3.1 eliminates at least $2n_4 - 4$ bends, since the transformation $T_1$ is applicable to each vertex of degree 4 which is not the topmost or bottommost vertex-segment in the visibility representation $\Gamma$, see fig. 9.a. Step 3.3 removes at least one bend for each vertex of
degree 2 or 3 which is not the topmost or bottommost vertex-segment in \( \Gamma \), provided that this bend has not already been removed by step 3.1 or step 3.2, see fig. 9.b. Therefore, at least \( n_2 + n_3 + 2n_4 - 4 \) bends are removed during step 3. hence the node-cost of \( G' \) is at most \( n_3 + 2n_4 + 4 \). □

As discussed in the proof of theorem 2, there is an overhead of at most 2 bends for each of the topmost and bottommost vertex-segments. If \( G \) is not biconnected, there is also an overhead of either one or two bends for some blocks. If both the topmost and the bottommost vertex-segments of a block \( B \) are cutpoints of \( G \), then there is no overhead. Otherwise, exactly one of the topmost and bottommost vertex-segments of \( B \) can be a cutpoint of \( G \). In other words, \( B \) is connected with the rest of the graph at one vertex. Suppose that the topmost vertex-segment \( v \) of block \( B \) is not a cutpoint of \( G \). If \( v \) has degree four, then the overhead for this block is two. Otherwise, it is one if \( v \) has degree three. Clearly, there is no overhead if \( v \) has degree two.

**Theorem 3** Let \( G = (V, E) \) be an \( n \)-vertex 1-connected graph of degree less than or equal to four, and \( n_i \) be the number of vertices with degree \( i \). Let also \( b_3 \) and \( b_4 \) be the number of blocks of \( G \) that have maximum vertex degree 3 and 4, respectively. Then the algorithm GRID-EMBEDDING produces a grid embedding \( G' \) for \( G \) with node-cost \( \text{N}(G') \leq n_3 + 2n_4 + b_3 + 2b_4 + 2 \leq 2.5n + 2. \)

![Figure 9](image-url)
Proof: From the discussion above, it is clear that \( N(G') \leq n_3 + 2n_4 + b_3 + 2b_4 + 2 \). The quantity \( b_3 + 2b_4 \) is maximized when \( b_4 \) is maximum. Since each block with a vertex of degree four must have at least five vertices, it can be shown that \( b_4 \leq \frac{n}{4} \). Therefore, \( N(G') \leq 2.5n + 2 \). □

The embedding produced by our algorithm is also area-efficient:

**Theorem 4** The algorithm **GRID-EMBEDDING** produces a grid embedding of an \( n \)-vertex planar graph \( G \) that occupies \( O(n^2) \) area.

**Proof:** The algorithm **W-VISIBILITY** constructs a \( w \)-visibility representation of \( G \) that is of height at most \( n \) and of width at most the number of faces (i.e., \( n + 2 \)). The steps 2, 3, and 4 of the algorithm **GRID-EMBEDDING** can only compact the embedding. Therefore, the embedding of \( G \) produced by our algorithm occupies \( O(n^2) \) area. □

6. CONCLUSIONS

The problem of embedding a graph in the rectilinear grid is a very interesting problem in VLSI theory. Typically, we want to minimize the area occupied by the embedding, and the total number of bends along the edges.

In this paper, we have presented a new embedding algorithm that runs in linear time. This is the first linear time algorithm, to the best of our knowledge, that achieves the following desirable properties. All the grid embeddings produced by our algorithm have at most \( 2.5n + 2 \) bends, and occupy area \( O(n^2) \), which is the best possible in the worst case [19]. Therefore, our results constitute a significant improvement upon the previously known results of [1] as far as time complexity and the node-cost, and of [13] as far as time complexity and the area measures are concerned. We are currently working on improving the worst-case node-cost bounds. Finally, we would like to mention that it is an open problem whether there is a linear time algorithm that constructs embed-
dings which occupy $O(n^2)$ area and the total number of bends is minimum.

ACKNOWLEDGMENTS

We wish to thank Franco Preparata for his guidance and support during this research. We also thank Scot Hornick and Majid Sarrafzadeh for suggesting ways to improve the presentation.
REFERENCES


