# Compact Encodings of Planar Orthogonal Drawings*

Amrita Chanda Ashim Garg

Department of Computer Science and Engineering
University at Buffalo
Buffalo, NY 14260
{achanda,agarg}@cse.buffalo.edu

**Abstract**

We present time-efficient algorithms for encoding (and decoding) planar orthogonal drawings of degree-4 and degree-3 biconnected and triconnected planar graphs using small number of bits. We also present time-efficient algorithms for encoding (and decoding) turn-monotone planar orthogonal drawings.

## 1 Introduction

It is important to compress the representation of planar orthogonal drawings to reduce their storage requirements and transmission times over a network, like Internet. The encoding problem is also interesting from a theoretical viewpoint. We investigate the problem of encoding planar orthogonal drawings of degree-4 and degree-3 biconnected and triconnected planar graphs using small number of bits, and present several results.

Let $d$ be a planar orthogonal drawing, with $b$ turns (bends) of a plane graph $G$ with $n \geq 3$ vertices, $m$ edges, and $f$ internal faces. Suppose each line-segment of $d$ has length at most $W$.

Our results are summarized in the following table, which shows for various types of graphs, the lengths of the encodings of $d$, and the times required to construct these encodings, and to decode them to obtain $d$ again:

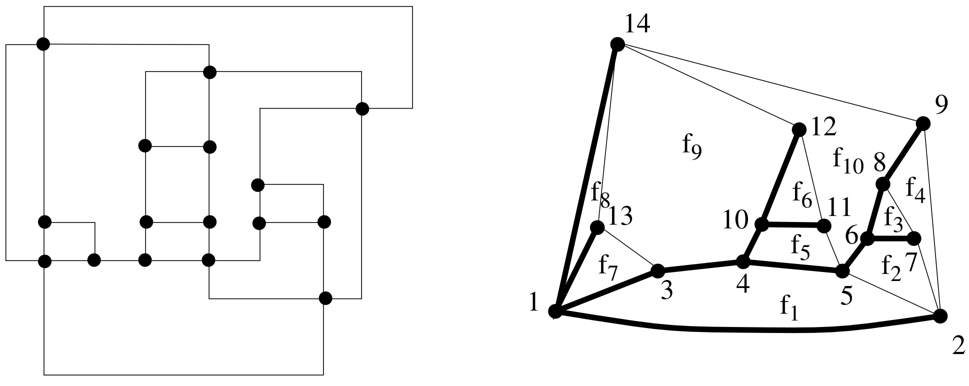
| Graph Type | Length of Encoding (in bits) | En(De)coding Time |
|---|---|---|
| Degree-4 Biconnected | $4.74m + 2.42n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $5.67m + 2n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| Degree-4 Triconnected | $3.58m + 2.59n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $3.67m + 2.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| Degree-3 Biconnected | $4.74m + 1.23n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $3.16m + n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| Degree-3 Triconnected | $3.67m + n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |

Several drawing algorithms [6, 7], that try to minimize the number of bends, produce *turn-monotone* drawings, i.e., where for each edge $e = (u, v)$, if we travel from $u$ to $v$ along $e$, then we will either make left turns or right turns, but not both (see Figure 1(a)). Turn-monotone drawings are very common in practice. We show that such drawings can be encoded even more succinctly, as shown in the following table:

| Graph Type | Length of Encoding (in bits) | En(De)coding Time |
|---|---|---|
| Degree-4 Biconnected | $3.16m + 4n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $4m + 3.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| Degree-4 Triconnected | $2m + 4.17n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $2m + 4.34n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| Degree-3 Biconnected | $3.16m + 2.81n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $1.49m + 2.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| Degree-3 Triconnected | $2m + 2.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |

As a by-product, our technique also encodes orthogonal representations, which represent the shape of a drawing, and are important intermediate constructs used by several drawing algorithms [6, 3].

*Research supported by NSF CAREER Award IIS-9985136 and NSF CISE Research Infrastructure Award No. 0101244.




(a) (b)

**Figure 1:** (a) A turn-monotone planar orthogonal drawing. (b) corresponding plane graph and the rightmost canonical ordering $c$; each vertex is labeled by its number in $c$; edges of the canonical spanning tree are shown as dark lines; each face is also labeled by its number in corresponding ordering of faces.

# 2 Previous Related Work

We are not aware of any previous work on encoding of planar orthogonal drawings. However, a variety of work has been done on encoding planar graphs. Let $G$ be a planar graph with $n$ vertices and $m$ edges. It is known that If $G$ is a triconnected, then it can be encoded using at most $1.58(n+m)$ bits [1]. If $G$ is biconnected, then it can be encoded using at most $2n + 1.58m$ bits [1]. If $G$ is a triangulated graph,then it can be encoded using at most 1.33 bits [1], and Tutte showed that any encoding of $G$ requires at least $1.08m$ bits [8] ($G$ may contain multiple edges). In [4], a technique is presented for encoding $G$ in asymptotically the minimum number of bits in $O(n \log n)$ time. For more results on graph encoding, see [1].

Our encoding technique is based on the graph encoding technique of [1], and on the concept of canonical orderings of planar graphs [2, 5, 1].

In Section 3, we give some definitions. In Sections 4, 5, and 6, we show how to encode (and decode) degree-3 and degree-4 plane graphs, orthogonal representations, and edge-lengths of the line-segments of an orthogonal drawing. In Section 7, we give our overall algorithm for encoding (and decoding) an orthogonal planar drawing.

# 3 Preliminaries

We use standard definitions of graph-theoretic terms. A *plane* graph is a planar graph equipped with an embedding. Let $u_1, u_2, \ldots, u_k$ be some vertices of a graph $G$. The graph *induced* by $u_1, u_2, \ldots, u_k$ is the maximal subgraph of $G$ that consists of these vertices and their incident edges. Suppose $G$ has $n$ vertices. An *ordering* $v_1, v_2, \ldots, v_n$ of the vertices of $G$ is an assignment of unique integer numbers in the range $[1, n]$ to the vertices of $G$, such that the $i^{th}$ vertex $v_i$ in the order is assigned number $i$.

Let $G$ be a degree-4 plane graph. Two planar orthogonal drawings of $G$ are *shape equivalent* when: (1) for each vertex $v$, consecutive edges incident to $v$ form the same angle at $v$ in the two drawings, and (2) for each edge $(u, v)$, the sequence of left and right turns encountered while walking from $u$ to $v$ following the polygonal chain representing $(u, v)$ is the same in the two drawings. An *orthogonal representation* $\Gamma$ of $G$ describes a class of shape equivalent planar orthogonal drawings of $G$. $\Gamma$ is a *turn monotone* representation if each edge is represented as a polygonal chain consisting of only left or right turns, but not both (see Figure 1(a)).

An important concept used by our encoding technique is the *canonical ordering* for plane graphs (see Figure 1(b)). This concept has been defined and used in [2, 5, 1]. Let $G = (V, E)$ be a simple biconnected plane graph with $n$ vertices, and $m$ edges, where $n \geq 3$. Let $v_1, v_2, \ldots, v_n$ be an ordering of the vertices of $G$. Let $G_i$ be the plane graph induced by vertices $v_1, v_2, \ldots, v_i$. Let $H_i$ be the external face of $G_i$.

**Definition 1 ([1])** *Let $v_1, v_2, \ldots, v_n$ be an ordering of the vertices of a biconnected plane graph $G = (V, E)$, where $v_1$ and $v_2$ are arbitrary two vertices on the external face of $G$ with $(v-1, v_2) \in E$. The ordering is* canonical *if there exist ordered partitions* $I_1, I_2, \ldots, I_K$ *of the interval* $[3, n]$ *such that the following properties hold for every* $1 \leq j \leq K$*: Suppose* $I_j = [k, k+q]$*. Let* $C_j$ *be the path* $(v_k, v_{k+1}, \ldots, v_{k+q})$ *(Note,* $C_j$ *is a single vertex if* $q = 0$*). A vertex* $u$ *of* $G_{k-1}$ *is a* neighbor *of* $C_j$ *if a vertex of* $C_j$ *is a neighbor of* $u$*.*

- *The graph* $G_{k+q}$ *is biconnected. Its external face* $H_{k+q}$ *contains the edge* $(v_1, v_2)$*, and the path* $C_j$*.* $C_j$ *has no chord in* $G$*, i.e.,* $G$ *does not contain any edge* $(v_s, v_t)$*, if* $|s-t| > 1$ *and* $k \leq s, t \leq k+q$*.*
- $C_j$ *has at least two neighbors in* $G_{k-1}$*, all of which are vertices of* $H_{k-1}$*. The leftmost neighbor is a neighbor of* $v_k$ *and the rightmost neighbor is a neighbor of* $v_{k+q}$*. Moreover, if* $q > 0$*, then* $v_l$ *and* $v_r$ *are the only neighbors of* $C_j$ *in* $G_{k-1}$*. The leftmost and rightmost neighbors of* $C_j$ *in* $G_{k-1}$ *are defined as follows: Vertices* $v_1$ *and* $v_2$ *divide* $H_{k-1}$ *into two paths: a path consisting only of edge* $(v_1, v_2)$*, and another path* $P = v_1(=u_1)u_2 \ldots u_s(=v_2)$ *that connect* $v_1$ *and* $v_2$*, and that does not contain the edge* $(v_1, v_2)$*. A vertex* $u_l$ *(*$u_r$*) of* $P$ *is the* leftmost *(*rightmost*) neighbor of* $C_j$ *in* $G_{k-1}$ *if* $u_l$ *is a neighbor of* $C_j$*, and there is no other vertex* $u_t$ *in* $P$ *such that* $u_t$ *is a neighbor of* $C_j$ *and* $t < l$ *(*$t > r$*).*

The definition of a *canonical* ordering for a triconnected plane graph is the same as that of the canonical ordering for a biconnected plane graph, except that it has the following additional property:

**Property 1 ([1])** *Every vertex* $v_k$*, where* $1 \leq k \leq n-1$ *has at least one neighbor* $v_p$ *with* $j < p$*.*

A *rightmost canonical (rmc) ordering* for a biconnected plane graph $G$ is defined as follows (see Figure 1(b)):

**Definition 2 ([1])** *Let* $v_1, v_2, \ldots, v_n$ *be a canonical ordering for* $G$*, where* $I_1, I_2, \ldots, I_K$ *are its corresponding interval partitions. We say that* $v_1, v_2, \ldots, v_n$ *is a rightmost canonical (rmc) ordering for* $G$ *is the following property holds for every interval* $I_j$*, where* $1 \leq j \leq K$*:*
*Suppose* $I_j = [k, k+q]$*. Let* $v_1, v_2, \ldots, v_{k-1}, u_k, u_{k+1}, \ldots, u_n$ *be any canonical ordering for* $G$ *whose first* $j-1$ *interval partitions are exactly* $I_1, I_2, \ldots, I_{j-1}$ *(Clearly, the* $G_{k-1}$ *and* $H_{k-1}$ *with respect to both canonical orderings are the same). Let* $v_l$ *be the leftmost neighbor of* $v_k$ *on* $H_{k-1}$*. Let* $u_l$ *be the leftmost neighbor of* $u_k$ *on* $H_{k-1}$*. Then,* $v_l$ *is to right of* $u_l$ *on* $H_{k-1}$*.*

**Theorem 1** *[1] Every biconnected plane graph* $G$ *with* $n$ *vertices admits a rightmost canonical (rmc) ordering. Moreover, a rightmost canonical ordering of* $G$ *can be constructed in* $O(n)$ *time.*

Any canonical ordering $c = v_1, v_2, \ldots, v_n$ of a biconnected plane graph $G$ and its corresponding interval partitions $I_1, I_2, \ldots, I_K$ defines a *canonical spanning tree* $T_c$ that consists of the edge $(v_1, v_2)$ plus the union of the paths $v_l v_k v_{k+1} \ldots v_{k+q}$ over all intervals $I_j = [v_k, v_{k+q}]$, where $1 \leq j \leq K$ and $v_l$ the leftmost neighbor of $v_k$ on $H_{k-1}$. Suppose we root $T_c$ at $v_1$. We can categorize the edges incident on a vertex $v$ as *tree* and *non-tree* edges: *Tree* edges are those that also belong to $T_c$, and *non-tree* edges are the remaining edges. Among the tree edges, let us call as *incoming* tree edges, the edges that connects $v$ with its the parent in $T_c$, and the edges that connect $v$ with its children as the *outgoing* tree edges. Suppose vertex $v$ belongs to interval $I_j = [v_k, v_{k+q}]$. The *incoming* non-tree edges of $v$ are those that connect $v$ to vertices in $G_{k-1}$, and the *outgoing* non-tree edges are the remaining non-tree edges. Note that by the definition of canonical ordering, each outgoing non-tree edge of $v$ will be of the form $(v, v_s)$, where $s > k+q$.

The following Properties follow easily from the definitions of canonical orderings for biconnected and triconnected graphs:

**Property 2** *Let* $v \neq v_1$ *be a vertex of* $G$*. Then,* $v$ *has exactly one non-tree edge, and either at least one outgoing tree-edge, or at least one incoming non-tree edge, or both. Vertex* $v_1$ *has only outgoing tree edges.*

**Property 3** *If* $G$ *is a triconnected graph, then for every vertex* $v \neq v_1, v_n$ *of* $G$*,* $v$ *has Property 2. Also, it has either at least one outgoing tree-edge, or at least one outgoing non-tree edge. Vertex* $v_1$ *has only outgoing tree edges. Vertex* $v_n$ *has exactly one incoming tree edge and all its other edges are incoming non-tree edges.*

The following Theorem can be derived easily from the results of [1].

**Theorem 2** *Let* $G$ *be a biconnected plane graph with* $n$ *vertices. Suppose, we are given a rightmost canonical ordering* $c = v_1, v_2, \ldots, v_n$ *of the vertices of* $G$*, along with the number of outgoing tree edges, incoming non-tree edges, and outgoing non-tree edges of each vertex as defined by* $c$*. Then, we can determine in* $O(n)$ *time, all the edges in* $G$*, as well as its embedding. In other words, given this information, we can determine the entire structure of graph* $G$*.*

## 4 Encoding Degree-3 And Degree-4 Plane Graphs

The algorithms given in [1] will encode a biconnected (triconnected) degree-4 plane graph using $5.17n$ bits (4.74 bits), and a degree-3 biconnected (triconnected) plane graph using $4.37n$ bits ($3.95n$ bits). But, the algorithms of [1] do not consider the degrees of vertices while encoding a graph. The algorithm of [4] will construct an asymptotically bit-minimum encoding of these graphs, but it is practical only for very large graphs. Here, we show that we can get a better encoding for degree-3 and degree-4 graphs by considering the degrees of their vertices.

The basic idea is very simple. Suppose we construct a rightmost canonical ordering $c = v_1, v_2, \ldots, v_n$ of the vertices of a biconnected plane graph $G$. Then, to encode $G$, from Theorem 2, it is sufficient to encode, for each vertex, how many outgoing tree edges, incoming non-tree edges, and outgoing non-tree edges the vertex has.

Suppose $G$ is a degree-3 graph. Let $v \neq v_1$ be a vertex of $G$. From Property 2, $v$ can only be of one of the following 7 types based on the number and type of its outgoing edges: (a) *Type A:* it has exactly two outgoing tree edges; (b) *Type B:* it one outgoing tree edge and one incoming non-tree edge; (c) *Type C:* it one outgoing tree edge and one outgoing non-tree edge; (d) *Type D:* it has one incoming non-tree edge, and one outgoing non-tree edge; (e) *Type E:* it has two incoming non-tree edges; (f) *Type F:* it has one incoming tree edge and one incoming non-tree edge; and (g) *Type G:* it has one incoming tree edge and one outgoing tree edge. Note that vertex $v_1$ either will have two outgoing tree edges or three outgoing tree edges. Thus, we encode $G$ by a string $S = s_1 s_2 \ldots s_n$, where

- $s_1$ represents the number of outgoing tree edges of $v_1$, and is equal to 0 if $v_1$ has two outgoing edges, and is equal to 1 if $v_1$ has three outgoing edges.
- Each symbol $s_i$, $2 \leq i \leq n$, represents the type of vertex $v_i$, and is equal to $A$, $B$, $C$, $D$, $E$, $F$, or $G$.

Since each $s_i$, where $2 \leq i \leq n$, can have 7 possible values, we can encode the substring $S' = s_2 \ldots s_n$ using $(n-1)\log_2 7 = 2.81(n-1)$ bits by converting the corresponding Base-7 number into binary representation in $O(n^2)$ time. Using Huffman encoding, we can encode $S'$ using at most $3(n-1)$ bits in $O(n)$ time. This, combined with Theorems 1 and 2, gives us the following lemma:

**Lemma 1** *Given a degree-3 biconnected graph $G$ with $n \geq 3$ vertices, we can encode it using less than $2.81n$ bits in $O(n^2)$ time and decode the encoding to reconstruct $G$ in $O(n^2)$ time. We can also encode $G$ using at most $3n-2$ bits and decode the encoding to reconstruct $G$ in $O(n)$ time.*

If $G$ is a triconnected degree-3 graph, then we can obtain an even shorter encoding for $G$. Let $v \neq v_1, v_n$ be a vertex of $G$. From Property 3, $v$ can only be of one of the following 4 types: (a) *Type A:* it has two outgoing tree edges; (b) *Type B:* it has one outgoing tree edge and one incoming non-tree edge; (c) *Type C:* it has one outgoing tree edge and one outgoing non-tree edge; and (d) *Type D:* it has one incoming non-tree edge, and one outgoing non-tree edge. Note that vertex $v_1$ will have exactly three outgoing tree edges, and $v_n$ will have exactly one incoming tree edge, and two incoming non-tree edge. So, we do not need to encode $v_1$ and $v_n$. We have the following lemma:

**Lemma 2** *Given a degree-3 triconnected graph $G$ with $n \geq 3$ vertices, we can encode it using at most $2n-2$ bits in $O(n)$ time (using Huffman Encoding). This encoding can be decoded in $O(n)$ time to reconstruct $G$.*

If $G$ is a degree-4 biconnected graph then each vertex $v$, where $v \neq v_1$, can be of 16 types. Therefore:

**Lemma 3** *Given a degree-4 biconnected graph $G$ with $n \geq 3$ vertices, we can encode it using at most $4n-2$ bits in $O(n)$ time (using Huffman Encoding) and decode the encoding to reconstruct $G$ in $O(n)$ time.*

If $G$ is a degree-4 triconnected graph then each vertex $v$, where $v \neq v_1, v_n$, can be of 12 types. Therefore:

**Lemma 4** *Given a degree-4 triconnected graph $G$ with $n \geq 3$ vertices, we can encode it using at most $2 + (n-2)\log_2 12 + 1 < 3.59n$ bits in $O(n^2)$ time and decode the encoding to reconstruct $G$ in $O(n^2)$ time. We can also encode $G$ using at most $2 + 3.67(n-1) + 1 < 3.67n$ bits (using Huffman encoding) and decode the encoding to reconstruct $G$ in $O(n)$ time.*

## 5 Encoding An Orthogonal Representation

We will use the following properties of an orthogonal representation:

**Property 4** *Sum of angles around any vertex is equal to $360°$.*

**Property 5** *Sum of interior angles of the polygon p representing any internal face is equal to* $(k-2)180°$, *where k is the total number of line-segments in p.*

We can encode an orthogonal representation $\Gamma$ of a plane graph $G$ by:

- *encoding structure:* encoding the structure of graph $G$,
- *encoding angles:* encoding the angles made by consecutive edges incident on each vertex, and
- *encoding turns:* for each edge $(u, v)$, encoding the sequence of left and right turns encountered while walking from $u$ to $v$.

To encode angles, suppose $G$ is a biconnected graph with $n$ vertices, and $m$ edges, where $n \geq 3$. Each angle of $\Gamma$ is either $90°$, $180°$, or $270°$. Suppose we have already constructed a rightmost canonical ordering $c = v_1, v_2, \ldots, v_n$ of the vertices of $G$. Let $v_i$ be a vertex of $G$. Let $e_1 e_2 \ldots e_k$, where $k \leq 4$ be the counterclockwise order of edges incident on $v_i$, where, if $v_i \neq v_1$, then $e_1$ is the incoming tree edge of $v$, and if $v_i = v_1$, then $e_1$ is the edge $(v_1, v_2)$. Let $s_i^*$ be the string $a_1 a_2 \ldots a_k$, where $a_j$ represents the counterclockwise angle between edges $e_j$ and $e_{j+1}$ at vertex $v_k$. $a_j$ is equal to $A$, $B$, or $C$, respectively, if the magnitude of the angle is equal to $90°$, $180°$, or $270°$, respectively. Then, we can construct a string $S^* = s_1^* s_2^* \ldots s_n^*$, that encodes all the angles of $G$. Total number of symbols in $S^*$ is equal to number of angles in $\Gamma$, which is equal to $2m$.

Using Property 4 of orthogonal representations, we can encode $S^*$ using even fewer bits. Property 4 implies that, for each vertex $v_i$, it is sufficient to encode angles $e_1, e_2, \ldots e_{k-1}$ only since the value of angle $a_k$ can be obtained from them. Thus, for $v_i$, it is sufficient to construct the string $s_i^* = a_1 a_2 \ldots a_{k-1}$. Now, the overall number of symbols in string $S^*$ can be reduced to $2m - n$. We have the following lemma:

**Lemma 5** *Given an orthogonal representation of a degree-4 biconnected graph G with* $n \geq 3$ *vertices, we can encode its angles using at most* $(2m-n)\log_2 3 = 1.58(2m-n) \leq 4.74n$ *bits in* $O(n^2)$ *time and in* $2(2m-n) \leq 6n$ *bits in* $O(n)$ *time (using Huffman Encoding). More over, during decoding, if we already know the degree of each vertex, then we can decode these encodings to obtain the angles in* $O(n^2)$ *and* $O(n)$ *time, respectively.*

If $G$ is a triconnected graph, then each vertex has at least 3 angles around it, and so each angle can be either $90°$, or $180°$. Therefore:

**Lemma 6** *Given an orthogonal representation of a degree-4 triconnected graph G with* $n \geq 3$ *vertices, we can encode its angles using at most* $2m - n$ *bits in* $O(n)$ *time. More over, during decoding, if we already know the degree of each vertex, then we can decode the encoding to obtain the angles in* $O(n)$ *time.*

Now consider the problem of encoding turns of a biconnected plane graph $G$. Given a canonical ordering $c = v_1, v_2, \ldots, v_n$, and the associated canonical tree $T_c$, we can construct an ordering $o = e_1, e_2, \ldots, e_m$ of edges as follows: Direct each edge $e = (v_j, v_k)$ from $v_j$ to $v_k$, where $j < k$. Put the edges of the vertices $v_1, v_2, \ldots, v_n$ in $o$, such edges of $v_i$ precede those of $v_j$ if $i < j$, and for each vertex $v_i$, we first put its incoming tree edge, followed by its incoming non-tree edges in the same order as their counter-clockwise order in $G$. Now, we construct a string $S^+ = s_1^+ \# s_2^+ \# \ldots \# s_n^+$, where each $s_i^+$ is a (possibly empty) sequence of symbols $L$ and $R$, and each symbol of $s_i^+$ corresponds to a turn of the (directed) edge $e_i = (v_j, v_k)$ encountered while going from $v_j$ to $v_k$ along edge $e_i$: symbol $L$ corresponds to a left turn and $R$ corresponds to a right turn. The turns are placed in $s_i^+$ in the order they are encountered.

**Lemma 7** *Given an orthogonal representation* $\Gamma$ *with b turns (bends) of a degree-4 biconnected graph G with* $n \geq 3$ *vertices, we can encode its turns using at most* $(b+m)\log_2 3 = 1.58(b+m)$ *bits in* $O(n^2)$ *time, and in* $1.67(b+m)$ *bits in* $O(n)$ *time (using Huffman Encoding). These encodings can be decoded in* $O(n^2)$ *and* $O(n)$ *time, respectively, to obtain the turns of* $\Gamma$.

We can further reduce the length of $S^+$ for turn-monotone orthogonal representations by using Property 5 of orthogonal representations. An interesting aspect of rightmost canonical ordering is that it can also be used to order the internal faces of a graph $G$, such that when we reconstruct the graph using the canonical ordering, starting from an initial graph consisting only of edge $(v_1, v_2)$, the faces get inserted into the graph in that order. Let $c = v_1, v_2, \ldots, v_n$ be a rightmost ordering of the vertices of a degree-4 biconnected plane graph $G$. Let $I_1, I_2, \ldots, I_K$ be the corresponding intervals of $c$. $c$ induces an ordering $f_1, f_2, \ldots, f_p$ of the internal faces of $G$ as follows: Let $I_1 = [v_3, v_{3+q'}]$. Face $f_1$ is the face consisting of the vertices $v_1, v_3, \ldots, v_{3+q'}, v_2$. In general, suppose we have already constructed the partial ordering $f_1, f_2, \ldots, f_s$ of the faces, using intervals

$I_1, I_2, \ldots, I_{k-1}$. Let $I_k = [v_k, v_{k+q}]$, where $q \geq 0$. Let $P = v_1(= u_1)u_2 \ldots u_s(= v_2)$ be the subpath of $H_{k-1}$ that we obtain by removing the edge $(v_1, v_2)$ from $H_{k-1}$. Let $C_j$ be the path $v_k v_{k+1} \ldots v_{k+q}$. We have two cases:

- $q > 0$: Then, by definition of canonical ordering, $C_j$ has exactly two neighbors $v_l$ and $v_r$ in $H_{k-1}$. Let $x_1 = (v_l)x_2, \ldots, x_t(= v_r)$ be the subpath of $P$ that connects $v_l$ and $v_r$. Then, $f_{s+1}$ is the internal face of $G$ consisting of the vertices $v_l(= x_1), v_k, \ldots, v_{k+q}, v_r(= x_t), x_{t-1}, x_{t-2}, \ldots, x_2$. We say that face $f_{s+1}$ *belongs to* Interval $I_k$.
- $q = 0$: Then $C_j$ consists of exactly one vertex $v_k$. Suppose we say that a vertex $u_i$ of $P$ is left (right) of another vertex $u_z$ of $P$, if $i < z$ $(i > z)$. Defining left and right in this fashion, suppose $u'_1(= v_l), u'_2, \ldots, u'_t(= v_r)$ be the left-to-right order of the neighbors of $v_k$ in $H_{k-1}$. Let $P'_i$, where $1 \leq i \leq t-1$, be the subpath of $P$ that connects vertices $u'_i$ and $u'_{i+1}$. Then, each face $f_{s+i}$, where $1 \leq i \leq t-1$, is the internal face that consists of the vertex $v_k$ and the vertices of path $P'_i$. We say that face $f_{s+i}$ *belongs to* Interval $I_k$.

Figure 1(b) shows the ordering of the faces. Suppose $T_c$ is the canonical spanning tree associated with $c$. For each face $f_i$ of $G$, a *tree* (*non-tree*) edge of $f_i$ is one that is also an edge of $T_c$. Another interesting fact is this:

**Fact 1** *Except for one non-tree edge $e$, all the non-tree edges of each face $f_i$ are already contained in the faces $f_1, f_2, \ldots, f_{i-1}$. We will call edge $e$ as the* non-tree completion edge *of $f_i$.*

Intuitively, we call the edge non-tree completion edge because, while reconstructing $G$ using $c$, this is the only non-tree edge that we need to add to the already constructed graph to add face $f_i$ to it (of course, we will need to add the tree edges of $f_i$ also). For example, in Figure 1, edge $(14, 12)$ is the non-tree completion edge of face $f_1 0$. For the face $f_s$, in the case $q > 0$ given above, the non-tree completion edge is $(v_{k+q}, v_r)$. For each face $f_{s+i}$, in the case $q = 0$ given above, the non-tree completion edge is $(v_k, u_{i+1})$.

Since each edge of a turn-monotone orthogonal representation $\Gamma$ has same kinds of turns only (left or right, but not both), Property 5 implies that for $\Gamma$, for any face $f$, it is sufficient to encode the turns of all but one edge $e$, since the turns of $e$ can be deduced from the turns of the other edges. In fact, following lemma says that it is sufficient to encode turns of tree edges:

**Lemma 8** *Let $\Gamma$ be an turn-monotone orthogonal representation of a degree-4 biconnected plane graph $G$. let $c$ be a rightmost canonical ordering of $G$. Suppose we construct a string $S^+$ encoding the turns of $\Gamma$ as in Lemma 7 using $c$, except that $S^+$ encodes the turns of* only *the tree edges of $G$. Then, by decoding $S^+$ we can obtain the turns of* all *the edges of $\Gamma$.*

**Proof:** Let $f_1, f_2, \ldots, f_p$ be the ordering of faces that corresponds to $c$, as defined above. We can easily prove this lemma can using induction:

*Base Case:* Consider face $f_1$. Decoding $S^+$ will give us the turns of all the tree edges of $f_1$. $f_1$ has exactly one non-tree edge $e$ (which is its non-tree completion edge). From Property 5, we can determine the turns of $e$ also.

*Induction*: Suppose we have already determined the turns of all the edges of faces $f_1, f_2, \ldots, f_{i-1}$. Consider face $f_i$. From Fact 1, except for its non-tree completion edge $e$, all the other non-tree of $f_i$ are already contained in the faces $f_1, f_2, \ldots, f_{i-1}$. Decoding $S^+$ will give us the turns of all the tree edges of $f_1$. Hence, except for $e$, we would know the turns of all the edges of $f_i$. From Property 5, we can determine the turns of $e$ also.

□

Since, $T_c$ has exactly $n-1$ edges, we have:

**Lemma 9** *Given a turn-monotone orthogonal representation $\Gamma$ with $b$ turns (bends) of a degree-4 biconnected graph $G$ with $n \geq 3$ vertices, we can encode its turns using at most $(b+n-2)\log_2 3 < 1.58(b+n)$ bits in $O(n^2)$ time, and at most $1.67(b+n)$ bits in $O(n)$ time (using Huffman Encoding). These encodings can be decoded in $O(n^2)$ and $O(n)$ time, respectively, to obtain all the turns of $\Gamma$.*

To encode an orthogonal representation, we construct a string $S_1 = AL'S'S^*S^+$, where $S'$, $S^*$, $S^+$ are strings encoding structure, angles, and turns, respectively, of $\Gamma$, as given by Lemmas 1 (or 2, 3, or 4), 5 (or 6), and 7 (or 9), respectively, $L'$ is length of $S'$ in binary notation, and $A$ encodes the length of $L'$ in unary, and consists of $|S'|$ 0's followed by a 1. Note that lengths of $A$ and $L'$ are $O(\log n)$ each.

## 6 Encoding Edge Lengths of A Planar Orthogonal Drawing

Let $d$ be a planar orthogonal drawing with $b$ turns (bends) of a degree-4 biconnected planar graph $G$ with $n \geq 3$ vertices and $m$ edges. Suppose each line-segment of $d$ has length at most $W$. Let $\Gamma$ be the orthogonal representation of $G$ that corresponds to $d$. Just as we encoded the turns of all the edges in a string $S^+$ in Section 5, we can construct a string $S' = s'_1 s'_2 \ldots s'_m$, where each $s_i$ contains the lengths of line-segments of edge $e_i = (v_j, v_k)$. The lengths are placed in the order the corresponding line-segments are encountered while traveling from $v_j$ to $v_k$, where $j < k$, along $e_i$.

We can reduce the length of $S'$ by making use of the following property of a planar orthogonal drawing: Suppose we orient each horizontal line-segment of $d$ as going "East" or "West", and each vertical line-segment as going "North" or "South", assuming that the line-segment of the edge $(v_1, v_2)$ incident on $v_1$ goes East. (This can be easily done in $O(n + b)$ time using the angle and turn information contained in $\Gamma$.)

**Property 6** *For each face $f_i$ of $G$, then in any planar orthogonal drawing $d$ of $G$:*

1. *Sum of the lengths of all the line-segments going East = the sum of the lengths of all the line-segments going West; and*
2. *Sum of the lengths of all the line-segments going North = the sum of the lengths of all the line-segments going South.*

Property 6 implies that we can omit encoding the length of one horizontal and one vertical line-segment of $f_i$, and still be able to obtain the lengths of all the line-segments of $f_i$ from an encoding of the lengths of its other line-segments. To decide, which line-segments to omit, consider the ordering $f_1, f_2, \ldots, f_m$ of the faces of $G$ that we can obtain from a rightmost ordering $c$ of $G$, as described in Section 5. Let $I_k = [v_k, v_{k+q}]$ be the interval of $c$, such that $f_i$ belongs to $I_k$. Let $E_i$ be the set of all the edges of $f_i$ that are not in the faces $f_1, f_2, \ldots, f_{i-1}$. Note that $E_i$ contains at least one edge, namely, the non-tree completion edge $e = (u, v)$ of $f_i$. Moreover, the edges of $E_i$ form a connected path $p$, which connects $u$ with a vertex $u'$, where $u$ is the end-vertex of $e$ that belongs to $H_{k-1}$, and $u'$ is a vertex common to both $f_i$ and $f_{i-1}$. We define the *free horizontal* (*vertical*) line-segment of $f_i$ to be the its first horizontal (vertical) line-segment encountered while traveling along $p$ from $u$ to $u'$. Note that $f_i$ will have at least one free line-segment (which can be horizontal or vertical). While encoding the lengths of the line-segments of $d$, we can omit from $S'$ the encodings of all the line-segments of $d$ that are free line-segments of the faces of $G$. We have the following lemma:

**Lemma 10** *We can encode the lengths of the line-segments of $d$ using a string $S'$ with $(\lfloor \log_2 W \rfloor + 1)(b + m - f_H - f_V) \leq (\lfloor \log_2 W \rfloor + 1)(b + m - f)$ bits in $O(n)$ time, where $f_H$ and $f_V$ are the number of horizontal and vertical free line-segments, respectively, of $d$. Assuming that, while decoding, we already know all the angles and turns of $d$, we can decode $S'$ to obtain the lengths of all the line-segments of $d$ in $O(n)$ time.*

## 7 Encoding A Planar Orthogonal Drawing

Let $d$ be a planar orthogonal drawing of a degree-4 biconnected planar graph $G$. Let $\Gamma$ be the orthogonal representation of $G$ that corresponds to $d$.

We can encode $d$ by constructing a string $S = BLS_1S_2$, where $S_1$ is the string constructed in Section 5 that encodes $\Gamma$, $S_2$ is the string constructed using Lemma 10 that encodes the free lengths of the line-segments of $d$, $L$ is a string, with length $\lfloor \log_2 |S_1| \rfloor + 1$, that encodes in binary notation the length of string $S_1$, and $B$ is a string that contains a sequence of $|L|$ 0's followed by a 1. $B$ encodes the length of $L$ in unary notation.

We can obtain $d$ by decoding $S$, by first extracting $A$ from it and obtaining the length of $L$, then extracting $L$, and obtaining the length of $S_1$, then extracting $S_1$ and decoding it to obtain $\Gamma$, and finally, extracting $S_2$ and decoding it to obtain the lengths of the line-segments of $d$. This is summarized in the following theorem:

**Theorem 3** *Let $d$ be a planar orthogonal drawing, with $b$ turns (bends) of a plane graph $G$ with $n \geq 3$ vertices, $m$ edges, and $f$ internal faces. Suppose each line-segment of $d$ has length at most $W$. Then the following table summarizes, for various types of graphs, the lengths of the encodings of $d$, and the times required to construct these encodings, and to decode them to again obtain $d$:*

| *Graph Type* | *Length of Encoding (in bits)* | *En(De)coding Time* |
|---|---|---|
| *Degree-4 Biconnected* | $4.74m + 2.42n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $5.67m + 2n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Degree-4 Triconnected* | $3.58m + 2.59n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $3.67m + 2.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Degree-3 Biconnected* | $4.74m + 1.23n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $3.16m + n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Degree-3 Triconnected* | $3.67m + n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Moreover, if d is a Turn-Monotone Drawing, then we can encode it using fewer bits, as follows:* | | |
| *Degree-4 Biconnected* | $3.16m + 4n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $4m + 3.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Degree-4 Triconnected* | $2m + 4.17n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $2m + 4.34n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Degree-3 Biconnected* | $3.16m + 2.81n + 1.58b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n^2)$ |
| | $1.49m + 2.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |
| *Degree-3 Triconnected* | $2m + 2.67n + 1.67b + (\lfloor \log_2 W \rfloor + 1)(b + m - f) + O(\log n)$ | $O(n)$ |

## References


[1] R. C. Chuang, A. Garg, X. He, M. Y. Kao, and H. Lu. Compact encoding of planar graphs via canonical ordering and multiple parenthesis. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science*, pages 118–129, 1998.

[2] H. de Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

[3] G. Di Battista, G. Liotta, and F. Vargiu. Spirality and optimal orthogonal drawings. *SIAM J. Comput.*, 27(6):1764–1811, 1998.

[4] Xin He, M.-Y. Kao, and H. Lu. A fast general methodology for information-theoretically optimal encodings of graphs. *SIAM J. Comput.*, 30(3):838–846, 2000.

[5] G. Kant. Drawing planar graphs using the *lmc*-ordering. In *Proc. 33th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 101–110, 1992.

[6] R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.

[7] R. Tamassia and I. G. Tollis. Planar grid embedding in linear time. *IEEE Trans. Circuits Syst.*, CAS-36(9):1230–1234, 1989.

[8] W. T. Tutte. A census of planar triangulation. *Canad. J. Math.*, 14:21–38, 1962.