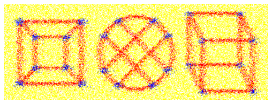

Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 0, no. 0, pp. 0–0 (0)

More Canonical Ordering

Melanie Badent Ulrik Brandes Sabine Cornelsen

Department of Computer & Information Science
University of Konstanz
Box 67, 78457 Konstanz, Germany

Abstract

Canonical ordering is an important tool in planar graph drawing and other applications. Although a linear-time algorithm to determine canonical orderings has been known for a while, it is rather complicated to understand and implement, and the output is not uniquely determined. We present a new approach that is simpler and more intuitive, and that computes a newly defined leftist canonical ordering of a triconnected graph which is a uniquely determined leftmost canonical ordering. Further, we discuss duality aspects and relations to Schnyder woods.

Submitted:	Reviewed:	Revised:	Reviewed:
December 2009	August 2010	September 2010	
Revised:	Accepted:	Final:	Published:
Article type: Regular paper		Communicated by:	

E-mail addresses: melanie.badent@uni-konstanz.de (Melanie Badent) ulrik.brandes@uni-konstanz.de (Ulrik Brandes) sabine.cornelsen@uni-konstanz.de (Sabine Cornelsen)

1 Introduction

Canonical vertex orderings were introduced by de Fraysseix, Pach, and Pollack [13, 14] and are the backbone of several algorithms for planar graphs.

De Fraysseix, Pach, and Pollack [13, 14] introduce an $\mathcal{O}(n \log n)$-time algorithm that embeds a maximal planar graph with n vertices on a $(2n-4) \times (n-2)$ integer grid by computing a canonical ordering of the vertices and then inserting them on a grid using this ordering. Later, Chrobak and Payne show how to execute this algorithm in linear time [10].

Kant [28] generalizes canonical orderings to triconnected graphs and presents a linear-time algorithm to construct a straight-line convex grid embedding of a triconnected graph on a $(2n-4) \times (n-2)$ grid. This grid size is improved by Chrobak and Kant [8] to $(n-2) \times (n-2)$. More about grid embeddings that are associated with canonical orderings can be found in [4, 8, 9, 10, 13, 14, 21, 22, 26, 27, 28]. Further, canonical vertex orderings are used to create visibility representations [20, 29, 30, 31], curve embeddings [17, 18], and other drawings [2, 3, 4, 16, 19, 20, 21, 22]. They are found in graph encoding [1, 11, 25], used for the construction of realizers, spanners, or orderly spanning trees [5, 6, 7, 15, 32, 33, 34], and more [12, 24, 36].

Kant [28] shows constructively that every triconnected planar graph has a canonical ordering and presents a linear-time algorithm. While several implementations of this algorithm of Kant are available, it is neither trivial to code, nor is its correctness easily understood. Based on a simple and intuitive criterion, we present a new algorithm that might further broaden the scope of adoption and ease teaching.

Since a triconnected graph can have many canonical orderings, we introduce the leftist (and rightist) canonical ordering that is uniquely determined. The leftist canonical ordering is in particular a leftmost canonical ordering.

The main advantage of our algorithm compared to the one in [28] is that we do not use the dual graph nor any face labels. Further, we compute the unique leftist canonical ordering from scratch, i. e., without any reordering, and we compute it from the low numbers to the high numbers contrary to the previous algorithm that builds the canonical ordering from the end by shelling off paths from the outer face. A similar approach for biconnected canonical orderings can be found in [23]. We also give a detailed pseudocode such that it can be easily implemented. Our proof of correctness includes a new proof of the existence of a canonical ordering for triconnected graphs. Finally, we show that the leftist canonical ordering induces the leftist canonical ordering of the dual graph.

Schnyder [34] develops the concept of Schnyder labelings (normal labelings) and Schnyder woods (realizers) for triangulated graphs. Di Battista, Tamassia, and Vismara [15] generalize Schnyder woods to triconnected graphs and show how to construct one from a canonical ordering. Felsner [**?**] constructs Schnyder labelings for triconnected graphs and proves that they are equivalent to Schnyder woods [**?**] and to α_0-orientations [**?**]. Recently, Gonçalves, Lévêque, and Pinlou [**?**] proved that there is a bijection between Schnyder woods and special contact triangle representations.

The minimal Schnyder wood is the Schnyder wood that is associated with the α_0-orientation without clockwise cycles [**?**]. Brehm [6] introduces an algorithm that directly constructs the minimal Schnyder wood for a triangulated graph. Fusy, Schaeffer, and Poulalhon [**?**] show how to construct the minimal α_0-orientation of a triconnected planar graph in linear time. Their algorithm works similar to the algorithm of Kant [28] for constructing a canonical ordering.

We discuss why there does not exist a reasonable one-to-one correspondence between canonical orderings and Schnyder woods and generalize this concept to ordered path partitions. Then, we give a bijection between the equivalence classes of ordered path partitions and Schnyder woods. Using the construction of Fusy, Schaeffer, and Poulalhon [**?**], we show that the leftist ordered path partition corresponds to the minimal Schnyder wood. We finally adapt our algorithm for the leftist canonical ordering such that it directly outputs the leftist ordered path partition.

The paper is organized as follows. Canonical orderings are defined in Section 2. The new algorithm and its linear-time implementation are described in Sections 3 and 4, respectively. In Section 5 we show how to find the leftist canonical ordering by the algorithm of Kant [28] and that the dual of the leftist canonical ordering corresponds to the leftist canonical ordering of the dual graph. Section 6 is dedicated to Schnyder woods and their bijection to equivalence classes of ordered path partitions.

2 Preliminaries

Throughout this paper, let $G = (V, E)$ be a simple undirected graph with n vertices, $n \geq 3$, and m edges. A graph G is *k-connected* if the removal of $k-1$ vertices does not disconnect the graph. A set of two vertices whose removal disconnects the graph is called a *separation pair*. We assume that G is planar and triconnected, hence it has a unique planar embedding up to the choice of the outer face.

For a subset $V' \subseteq V$ we denote by $G[V']$ the subgraph of G induced by V'. By $\deg_G(v)$ we denote the number of edges of G that contain v. A path is a sequence $P = \langle z_0, \ldots, z_\ell \rangle$ of distinct adjacent vertices, i. e., $\{z_i, z_{i+1}\} \in E$. We also denote the set $\{z_0, \ldots, z_\ell\}$ by P.

Canonical orderings were introduced originally for triangulated graphs by de Fraysseix et al. [13, 14]. The following rephrases Kant's generalization to triconnected graphs [28].

Definition 1 (canonical ordering) *Let $\Pi = (P_0, \ldots, P_s)$ be a partition of V into paths and let $P_0 = \langle v_1, v_2 \rangle$, $P_s = \langle v_n \rangle$ such that $\langle v_2, v_1, v_n \rangle$ is a path on the outer face of G in clockwise direction. For $k = 0, \ldots, s$ let $G_k = G[V_k] = (V_k, E_k)$ be the subgraph induced by $V_k = P_0 \cup \cdots \cup P_k$, let C_k be the outer face of G_k. Partition Π is a* canonical ordering *of (G, v_1) if for each $k = 1, \ldots, s-1$:*

1. *C_k is a simple cycle.*
2. *Each vertex z_i in P_k has a neighbor in $V \setminus V_k$.*

3. $|P_k| = 1$ *or* $\deg_{G_k}(z_i) = 2$ *for each vertex* z_i *in* P_k.

P_k *is called a* singleton *if* $|P_k| = 1$ *and a* chain *otherwise.*

A canonical ordering Π is refined to a *canonical vertex ordering* $v_1, \ldots, v_n$ by ordering the vertices in each $P_k, k > 0$, according to their clockwise appearance on C_k (see Figures 1(a)-1(c)).

The following observations help build an intuitive understanding of canonical orderings. Each path P_k encloses an interval of consecutive faces of G_k adjacent to C_{k-1} on the outside of G_{k-1}. This interval consists of exactly one face if P_k is a chain and of one or more faces if P_k is a singleton. Iterative application of Condition 2 guarantees that for each $z_i \in P_k$ there is a path to v_n in $G[V \setminus V_k] \cup \{z_i\}$, i. e., a path not using a vertex in $V_k \setminus \{z_i\}$.

We summarize the above observations in Propositions 2 and 3 of Lemma 1. Propositions 4 and 5 of Lemma 1 are part of Kant's original definition.

Lemma 1 *For* $k = 1, \ldots, s-1$*:*

1. P_k *has no chord.*
2. *For each vertex* z *in* P_k *there is a* z-v_n*-path* $z = z_{k_0}, \ldots, z_{k_p} = v_n$ *where each* z_{k_i} *is in* P_{k_i} *and* $k_i < k_j$ *for* $0 \leq i < j \leq p$*. Especially:*
   (a) $G[V \setminus V_k]$ *is connected.*
   (b) *If* $\deg_{G_k}(z) = 2$*, then* v is in C_k.
   (c) P_k *is on* C_k.
3. (a) *A singleton* P_{k+1} *and a path of* C_k *bound some faces or*
   (b) *a chain* P_{k+1} *and a path of* C_k *bound one face.*
4. G_k *is biconnected.*
5. *If* v, w *is a separation pair of* G_k*, then both are on* C_k.

Proof: The properties are directly implied by the fact that G is triconnected and by the definition of a canonical ordering. □

Remark 1 *Two incident faces of a triconnected planar graph share one vertex or one edge. Especially, no face has a chord.*

2.1 Leftmost Canonical Ordering

In general, a canonical ordering of (G, v_1) is not uniquely defined. Therefore, Kant [28] introduced a leftmost and rightmost canonical ordering of (G, v_1). Let $P_0, \ldots, P_k$ be a sequence of paths that can be extended to a canonical ordering of G. A path P of G is a *feasible candidate* for the step $k+1$ if also $P_0, \ldots, P_k, P$ can be extended to a canonical ordering. Let $v_1 = c_1, c_2, \ldots, c_q = v_2$ be the vertices from left to right on C_k. Let c_ℓ be the neighbor of P on C_k such that ℓ is as small as possible and let c_r be the neighbor of P on C_k such that r is as large as possible. We call c_ℓ the *left neighbor* and c_r the *right neighbor* of P.

Definition 2 (leftmost canonical ordering) *A canonical ordering $P_0, \ldots, P_s$ is called* leftmost (rightmost) *if for $k = 0, \ldots, s-1$ the following is true. Let c_ℓ be the left neighbor of P_{k+1} and let $P_{k'}$, $k+1 \leq k' \leq s$, be a feasible candidate for the step $k+1$ with left neighbor $c_{\ell'}$. Then either (1) $\ell \leq \ell'$ ($\ell \geq \ell'$) or (2) there is an edge between P_{k+1} and $P_{k'}$ (see Figure 1(b)).*

Note that once a canonical ordering is known a simple linear-time algorithm can be used to rearrange its paths so that it becomes leftmost [28]. Also note that Kant did not use Condition 2 of a leftmost canonical ordering in his definition, however, he used it in his reordering algorithm. More precisely, let $P_0, \ldots, P_s$ be a canonical ordering of (G, v_1) and let $e = \{u_1, u_2\}$ be an edge of G such that there are $k_1 < k_2$ with $u_i \in P_{k_i}, i = 1, 2$. Then e is an *outgoing edge* of u_1 and an *incoming edge* of u_2.

Kant [28] required that a leftmost canonical ordering may be constructed by reordering a given canonical ordering only if the incoming and outgoing edges are maintained.

While leftmost canonical orderings are particularly useful for many applications, we stress that the rearrangement is applicable to any canonical ordering and that a leftmost canonical ordering is only unique with respect to a given partition.

2.2 Leftist Canonical Ordering

In the leftist canonical ordering we add in each step the leftmost possible path where the choice is not only within an already given partition.

Definition 3 (leftist canonical ordering) *A canonical ordering $P_0, \ldots, P_s$ is called* leftist (rightist) *if for $k = 0, \ldots, s-1$ the following is true. Let c_ℓ be the left neighbor of P_{k+1} and let P be a feasible candidate for the step $k+1$ with left neighbor $c_{\ell'}$. Then $\ell \leq \ell'$ ($\ell \geq \ell'$) (see Figures 1(c) and 1(a)).*

Note that a feasible candidate for the step $k+1$ needs not to be a feasible candidate for the step $k+2$ anymore. Also note that the leftist canonical ordering is unique irrespective of a given partition and it is a leftmost canonical ordering. A similar concept related to Schnyder labelings without clockwise cycles was defined for triangulated graphs in [6].

Now, our goal is not only to simplify the computation of an initial canonical ordering but also to compute the leftist canonical ordering of (G, v_1).

3 New Algorithm

Starting from $P_0 = \langle v_1, v_2 \rangle$, we build the canonical ordering by adding $P_1, \ldots, P_s$ in this order. In step $k+1$, the "belt" around G_k, i. e., the sequence of vertices not in G_k that lie on faces of G incident to G_k is considered. Then, a candidate not causing any "self-intersection" within the belt is chosen. Before we give the details, we start with a recursive definition of which paths will be considered in the step $k+1$.

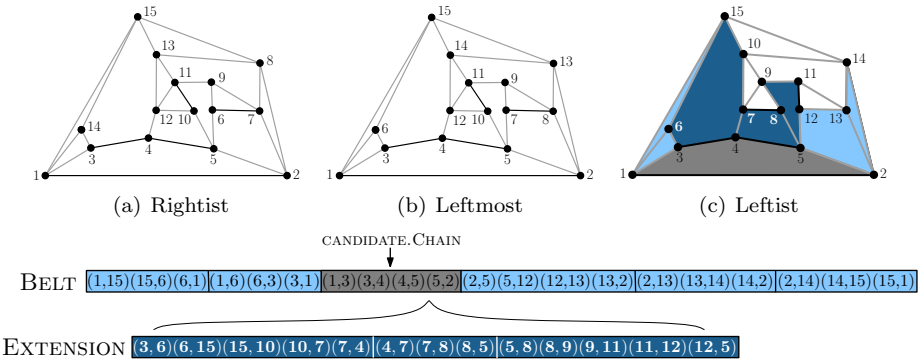


Figure 1: Different canonical orderings. Black paths are chains. In (c) the light blue faces and the gray face are the belt of G_0. The next candidate in Algorithm 3 is $P_1 = \langle 3, 4, 5 \rangle$. Algorithm 4 substitutes the gray face by the dark blue faces, i. e., by the EXTENSION found by Algorithm 5.

Definition 4 (cut faces and locally feasible candidates) *$P_0 = \langle v_1, v_2 \rangle$ is a locally feasible candidate. Let $P_0, \ldots, P_k$ be a sequence of locally feasible candidates and V_k, G_k, and C_k as in Definition 1. A* cut face *f of G_k is an inner face of G that is incident to some vertex on C_k but is not a face of G_k. Let P_f be the clockwise sequence of vertices incident to f that are not in V_k. If f is incident to an edge on C_k, then f is called a* candidate face *and P_f is called a* candidate *for the step $k+1$. A candidate face f and the candidate P_f are* locally feasible *for the step $k+1$ if*

1. *v_n is not in P_f or $P_0, \ldots, P_k, P_f$ is a partition of V,*
2. *$G[V \setminus (V_k \cup P_f)]$ is connected, and*
3. *P_f is a singleton or the degree of each vertex of P_f in $G[V_k \cup P_f]$ is two.*

In the remainder of this section, we will see that the locally feasible candidates are exactly the feasible candidates. We start with the following lemma which is a direct consequence of Definitions 1 and 4 and the triconnectivity of G.

Lemma 2

1. *A canonical ordering is a sequence of locally feasible candidates.*
2. *If a sequence of locally feasible candidates partitions the whole vertex set of a triconnected graph, it is a canonical ordering.*

Proof:

1. It follows directly from Definition 1 and Lemma 1(2a) that each canonical ordering is a sequence of locally feasible candidates.

2. Let $P_0, \ldots, P_s$ be a sequence of locally feasible candidates partitioning the whole vertex set of a triconnected graph $G = (V, E)$. We show by induction on $k = 0, \ldots, s$ that P_k fulfills the condition of a canonical ordering. $P_0 = \langle v_1, v_2 \rangle$ and v_n is in P_s. By triconnectivity and Condition 3 of a locally feasible candidate, P_s consists only of v_n.

   Let now $0 < k < s$. Since P_{k+1} is a candidate and the outer face of G_k had been a simple cycle by the inductive hypothesis, it follows that the outer face of G_{k+1} is a simple cycle. Condition 3 of a locally feasible candidate corresponds to Condition 3 of a canonical ordering.

   By triconnectivity of G, each vertex has at least degree 3. Hence, if P_{k+1} is a chain, each vertex of P_{k+1} is connected to $V \setminus V_{k+1}$. By the connectivity of $G[V \setminus V_k]$, a singleton has to be connected to some vertex in $V \setminus V_{k+1}$.

□

In what follows, we consider the vertices on C_k to be from left to right between v_1 and v_2. Accordingly, we also consider the cut faces from left to right: A *cut edge* of G_k is an edge of G that is incident to one vertex in V_k and one vertex in $V \setminus V_k$. Let f and f' be two cut faces. Let c and c', respectively, be the leftmost vertices on C_k that are incident to f and f', respectively. We say that f is to the left of f' if c is to the left of c' on C_k or if $c = c'$, then the cut edges of f are to the left of the cut edges of f' in the incidence list of c.

Algorithm 1: Leftist Canonical Ordering

```
begin
    Let v_2, v_1, v_3, ..., v_p be the bound of the inner face incident to {v_1, v_2}
    P_0 ← ⟨v_1, v_2⟩, P_1 ← ⟨v_3, ..., v_p⟩, k ← 1
    while |V_k| < n − 1 do
        Let f be the leftmost locally feasible candidate face
        P_{k+1} ← P_f
        k ← k + 1
    P_{k+1} ← ⟨v_n⟩
```

Corollary 1 *If Algorithm 1 terminates, it computes the leftist canonical ordering of a triconnected planar graph.*

Before we prove that in each step there exists a locally feasible candidate face, we describe locally feasible candidates in terms of "self-intersection" of the belt. Let $P_0, \ldots, P_k$ be a sequence of locally feasible candidates. The *belt* of G_k is the sequence of vertices not in G_k that are incident to the cut faces of G_k from left to right. I. e., let $f_1, \ldots, f_t$ be the cut faces of G_k ordered from left to right. Let P_{f_0} be the vertices in $V \setminus V_k$ that are incident to the outer face in counterclockwise order. Then, the concatenation of $P_{f_1}, \ldots, P_{f_t}$ and P_{f_0} is the belt of G_k. Consider Figure 2. Then, $P_2 = \langle 6, 7 \rangle$, $P_3 = \langle 8 \rangle$, and the belt of G_3 is $15, 14|14|14, 15, 13, 12|12, 10|10, 11, 9|9|9, 11, 13|13, 15|15$.

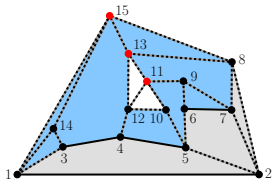


Figure 2: Rightist canonical ordering: G_3 is gray, cut faces are blue, red vertices are forbidden.

Definition 5 (forbidden, singular, stopper) *A vertex v of the belt of G_k is*

- forbidden *if v does not occur consecutively in the belt of G_k,*
- singular *if v occurs more than twice in the belt of G_k and its occurrence is consecutive, and*
- *a* stopper *if it is forbidden or singular.*

In the above example, 15, 13, and 11 are forbidden vertices and 14 and 9 are singular vertices. Note that v_n is always the first and last vertex of the belt. Hence, it remains forbidden until the end. It will turn out that the locally feasible candidates are those that do not contain a stopper or that are singular singletons.

Lemma 3 *Let $P_0, \ldots, P_k$ be a sequence of locally feasible candidates. Let f be a candidate face for the step $k+1$ and let $P = P_f$.*

1. *If a vertex v of P is adjacent to more than two vertices in $V_k \cup P$, then v occurs more than twice in the belt.*
2. *If $G[V \setminus (V_k \cup P)]$ is not connected, then P contains a forbidden vertex.*
3. *If a vertex v of P is singular, then v is a locally feasible singleton.*
4. *If P contains a forbidden vertex v, then $G[V \setminus (V_k \cup P)]$ is not connected or P contains another vertex with more than two neighbors in $V_k \cup P$.*

Proof:

1. First, assume $v \neq v_n$. Let e be an edge incident to v and a vertex in $V_k \cup P$ that is not incident to f. By Remark 1, edge e is a cut edge and hence incident to two cut faces. Thus, v is incident to at least three cut faces. If $v = v_n$, then v is the first and the last vertex of the belt and occurs also in f in the belt.
2. Let W be the set of vertices in a connected component of the graph induced by $V \setminus (V_k \cup P)$ and not containing v_n. Since $V \setminus V_k$ was connected, W is adjacent to P and there is a path from P to v_n not intersecting W. By the triconnectivity of G, there is an edge between W and the part of C_k not

contained in f. Further, there is at least a third vertex on $C_k \cup P$ adjacent to W. Let w be the rightmost vertex on $C_k \cup P$ that is adjacent to W and let v be the leftmost such vertex. Assume that w is on C_k. Then v is on P. Consider the face f' containing v and w. Then, the belt contains some vertices of W between the occurrences of v for the belt faces f and f' (see Figure 3(a)).

3. If v is singular, then it is a candidate. By Proposition 2, $G[V \setminus (V_k \cup \{v\})]$ is connected.

4. Since v is forbidden, there is a cut face f' containing v and a cut face h between f and f' such that P_h contains a vertex $w \neq v$. If w is not incident to f, then w and v_n are in two connected components of $G[V \setminus (V_k \cup P)]$ (see Figure 3(b)). So assume now that for all faces h' between f and f' the path $P_{h'}$ contains only vertices incident to f. Among these faces let h be the face that is next to f. By Remark 1, P_h consists of one vertex $w \neq v$ and w is singular (see Figure 3(c)).

□

Corollary 2

1. *A candidate that is a chain is locally feasible if and only if it does not contain any stopper.*

2. *A vertex of the belt is a locally feasible singleton if and only if it is singular.*

For example, the locally feasible candidates for the step $k + 1 = 4$ in Figure 2 are $\langle 14 \rangle$, $\langle 12, 10 \rangle$, and $\langle 9 \rangle$.

Theorem 1 *Algorithm 1 computes the leftist canonical ordering of a triconnected planar graph.*

Proof: By Lemma 1, it remains to show that in each step of the algorithm there is a locally feasible candidate. By Corollary 2(2), if there are any singular vertices, we have a locally feasible candidate. So, assume now we do not have any singular vertices. By Corollary 2, we have to show that there is a candidate that does not contain any forbidden vertex.

Let f be a candidate face and let $P = P_f$. Assume that P contains a forbidden vertex v. Let f' be a cut face containing v such that the belt contains a vertex other than v between the occurrence of v in P_f and the occurrence of v in $P_{f'}$. Let $f, h_1, \ldots, h_t, f'$ be the sequence of cut faces between f and f'. We show by induction on the number of forbidden vertices in $P_{h_1}, \ldots, P_{h_t}$ that there is a locally feasible candidate among $P_{h_1}, \ldots, P_{h_t}$.

By the choice of f' and by triconnectivity of G, there is at least one $i = 1, \ldots, t$ such that P_{h_i} is a candidate that does not contain v. If v is the only forbidden vertex in $P_{h_1}, \ldots, P_{h_t}$, then P_{h_i} is locally feasible.

If P_{h_i} contains a forbidden vertex w (recall that by our assumption there are no singular vertices), there is a cut face $h \neq h_i$ among $f, h_1, \ldots, h_t, f'$ incident

to w such that the belt contains a vertex other than w between the occurrence of w in P_{h_i} and in P_h. The cut faces between h and h_i do not contain v. Hence, by the induction hypothesis, one of them is a locally feasible candidate. □

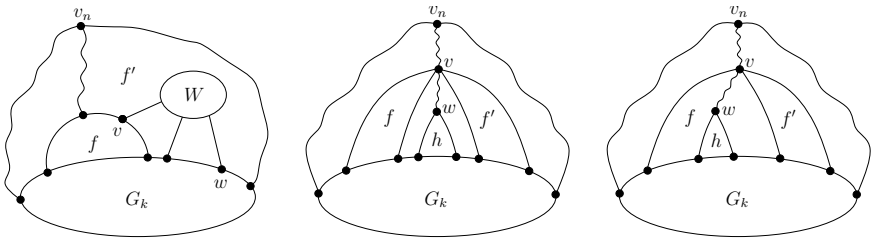


Figure 3: Illustration of the proof of Lemma 3. (a) W is a connected component of $G[V \setminus (V_k \cup P_f)]$ not containing v_n. Faces f and f' are not consecutive in the belt of G_k. Thus, f contains a forbidden vertex v. (b, c) If v is forbidden, then (b) $G[V \setminus (V_k \cup P_f)]$ is not connected or (c) there is a singular vertex w.

4 Linear-Time Implementation

We will maintain a list BELT that represents the cut faces from left to right. For a simpler implementation, BELT contains lists of edges rather than one list of vertices and each cut face f is represented by a *belt item* which is a pair consisting of

- a list CHAIN of f's incident edges not in G_k in clockwise order and
- the rightmost stopper of P_f (if any).

We traverse the list BELT using a pointer CANDIDATE.

To decide whether a vertex is a stopper, we maintain two counters. Let CUTFACES(v) be the number of cut faces and CUTEDGES(v) the number of cut edges to which v is incident. In order to make the following lemma true also for v_n, we will count the outer face twice in CUTFACES(v_n).

Lemma 4 *A vertex v in the belt of G_k is*

- *forbidden if and only if* CUTFACES(v) > CUTEDGES(v) + 1 *and*
- *singular if and only if* 2 < CUTFACES(v) = CUTEDGES(v) + 1.

Proof: A vertex occurs once for each cut face it is incident to in the belt. Two occurrences of a vertex v in the belt are consecutive if and only if the corresponding cut faces share a cut edge incident to v. So, all occurrences of v in the belt are consecutive if and only if v is only incident to one more cut face than to cut edges. □

Algorithm 2: Leftist Canonical Ordering

Input : $G = (V, E)$ planar embedded triconnected undirected graph
$v_1 \in V$ on the outer face
Output: leftist canonical ordering $P_0, \ldots, P_s$ of (G, v_1)

canonicalOrdering
replace each $\{v, w\} \in E$ by (v, w) and (w, v)
$v_n \leftarrow$ clockwise neighbor of v_1 on outer face
$v_2 \leftarrow$ counterclockwise neighbor of v_1 on outer face
for $v \in V$ **do** cutFaces$(v) \leftarrow 0$; cutEdges$(v) \leftarrow 0$
cutFaces$(v_n) \leftarrow 1$
mark (v_1, v_2) and (v_2, v_1)
Belt $\leftarrow \langle(\langle(v_2, v_1), (v_1, v_2), (v_2, v_1)\rangle,$ nil$)\rangle$
$k \leftarrow -1$
candidate $\leftarrow$ first item in Belt
while Belt $\neq \emptyset$ **do**
$k \leftarrow k + 1$
$P_k \leftarrow$ `leftmostFeasibleCandidate`
`updateBelt`

Algorithm 3: Skip infeasible candidates

list leftmostFeasibleCandidate
found $\leftarrow$ `false`
repeat
let $\langle(z_0, z_1), (z_1, z_2), \ldots, (z_p, z_{p+1})\rangle :=$ candidate.Chain
if $z_0 \neq z_{p+1}$ **then**
$j \leftarrow p$
$\blacktriangleright^1$ **while** $j > 0$ **and not**(*forbidden*(z_j) **or** *singular*(z_j)) **do**
$j \leftarrow j - 1$
if $j > 0$ **then** candidate.stopper $\leftarrow z_j$
$\blacktriangleright^2$ **if** $j = 0$ **or**(*singular*(candidate.stopper) **and** $p = 1$)
then
found $\leftarrow$ `true`
for $(v, w) \in$ candidate.Chain **do** mark (w, v)
if not found **then**
candidate $\leftarrow$ *successor*(candidate)
if candidate $=$ nil **then HALT: illegal input graph**
until found
return $\langle z_1, \ldots, z_p\rangle$

Algorithm 4: Replace feasible candidate with incident faces

```
updateBelt
    ▼ if singular(CANDIDATE.STOPPER) then
      remove neighboring items with same singleton from BELT
    PRED ← predecessor(CANDIDATE)
    SUCC ← successor(CANDIDATE)
    if SUCC ≠ ∅ then remove first edge from SUCC.CHAIN
    EXTENSION ← BeltExtension(CANDIDATE.CHAIN)
    replace CANDIDATE by EXTENSION in BELT
    if EXTENSION ≠ ∅ then
      CANDIDATE ← first item of EXTENSION
    else
      CANDIDATE ← SUCC
    if PRED ≠ ∅ then
        remove last edge (v, w) from PRED.CHAIN
        if v = PRED.STOPPER or w = source(first edge of PRED.CHAIN)
        then
            PRED.STOPPER ← NIL
            CANDIDATE ← PRED
```

Algorithm 5: Construct list of new belt items incident to P_k

```
list beltExtension(list ⟨e_0, ..., e_p⟩)
    EXTENSION ← ∅
    for j ← 1, ..., p do // scan for new cut faces incident to v_start
        v_start ← source(e_j)
        v_end ← target(e_j)
        FIRST ← e_j
        repeat
            FIRST = (v, w) ← clockwise next in N+(v_start) after FIRST
            CUTEDGES(w) ← CUTEDGES(w) + 1
            if FIRST not marked then                    //new cut face
                CHAIN ← ∅
                repeat                                  //traverse clockwise
                    mark (v, w)
                    append CHAIN ← (v, w)
                    CUTFACES(w) ← CUTFACES(w) + 1
                    (v, w) ← counterclockwise next in N+(w) after (w, v)
                until w ∈ {v_start, v_end}
                mark (v, w)
                append CHAIN ← (v, w)
                append EXTENSION ← (CHAIN, NIL)
        until w = v_end
    return EXTENSION
```

The algorithm `canonicalOrdering` (see Algorithm 2) now works as follows (for a detailed illustration see Figure 6). We start with a copy of G in which each undirected edge $\{v, w\}$ is replaced by the two directed edges (v, w) and (w, v). In the beginning, the belt is initialized by $(\langle(v_2, v_1), (v_1, v_2), (v_2, v_1)\rangle, \text{NIL})$. Thus, `leftmostFeasibleCandidate` (see Algorithm 3) chooses $P_0 = \langle v_1, v_2\rangle$ as the first path.

In general, each iteration in Algorithm 2 consists of three steps: (1) We choose the new leftmost locally feasible candidate P_k, (2) we find the new cut faces incident to P_k, and (3) we replace P_k by its incident cut faces in the belt and update its neighbors (see Figure 1(c)). In detail:

`leftmostFeasibleCandidate` We traverse BELT from the current cut face CANDIDATE to the right doing the following: If CANDIDATE is a candidate face, traverse CANDIDATE.CHAIN from right to left until a stopper is found. If so, store it. If CANDIDATE.CHAIN contains no stopper or it is a singular singleton, it is the next locally feasible candidate. Otherwise, go to the next face. See Algorithm 3.

`beltExtension` To find the new cut faces, we traverse CANDIDATE.CHAIN from left to right. The outer repeat-loop iterates over all vertices incident to two edges of CANDIDATE.CHAIN. Each iteration finds the new cut faces incident to such a vertex and increments the counter CUTEDGES. In the inner repeat-loop, we traverse all new edges of a new cut face and store them in the list CHAIN. Here the counter CUTFACES is incremented. Each list CHAIN is finally appended to the list EXTENSION that stores all new belt items incident to CANDIDATE.CHAIN. See Algorithm 5.

`updateBelt` We replace CANDIDATE (and all its copies if it was a singleton) by the new cut faces found by `beltExtension`. The last edge of the predecessor and the first edge of the successor of CANDIDATE are removed since they are now contained in G_k. If the predecessor of CANDIDATE was not a candidate face before or it lost its stopper, then we go one step to the left in BELT and set CANDIDATE to its predecessor. See Algorithm 4.

Theorem 2 *Algorithm 2 computes the leftist canonical ordering of a triconnected planar graph in linear time.*

Proof:

Linear running time: In the algorithm `beltExtension` each edge is touched at most twice. In the algorithm `leftmostFeasibleCandidate` each candidate is scanned from right to left until the first stopper occurs. All the scanned edges will have been deleted from the list when the candidate will be scanned the next time. In total only $2m$ edges will be added to BELT.

Correctness: While scanning BELT from left to right, we always choose the leftmost locally feasible candidate: Assume that at step $k + 1$ we scan a face f and there are no locally feasible candidates to the left of f. The

face f is omitted because it is not a candidate or it contains a stopper. None of the two properties changes if no direct neighbor in BELT had been added to G_k. Hence, as long as f is not locally feasible, no face to the left of f has to be considered. Further, the number of incident cut faces or cut edges of a vertex never decreases. We show that a candidate can only become locally feasible after his rightmost stopper has become singular.

Let v be the rightmost stopper of P_f and assume v is forbidden. Let f_ℓ be the leftmost and f_r be the rightmost cut face containing v. We can conclude from the proof of Theorem 1 that all occurrences of v between f_ℓ and f are consecutive and that Algorithm 2 finds the locally feasible candidates between f and f_r in the belt until the belt contains only v between f and f_r. It follows that all occurrences of v in the belt are consecutive. Let now v be singular. Then, the only two incident cut faces $f = f_\ell$ and f_r of v would share a cut edge $\{v, w\}$ that would not have been a cut edge before. Hence, w would have been a stopper of f to the right of v.

□

Note that the algorithm for computing the leftist canonical ordering can also be used to compute the rightist canonical ordering. In that case, we store for each cut face the leftmost stopper and we scan the belt from right to left.

5 Duality

In this section, we show that the leftist canonical ordering can also be found by choosing always the rightmost face or singleton in the algorithm of Kant [28]. We conclude that the dual of the leftist canonical ordering is the leftist canonical ordering of the dual graph.

Let $\langle v_2, v_1, v_n\rangle$ be a path on the outer face of G in clockwise direction. Let $\Pi = (P_0, \ldots, P_{k-1})$ with $P_0 = \langle v_1, v_2\rangle$, $P_{k-1} \neq \langle v_n\rangle$ or $\Pi = (P_{k+1}, \ldots, P_s)$ with $\langle v_1, v_2\rangle \neq P_{k+1}, P_s = \langle v_n\rangle$, respectively, be a sequence of paths of G that can be extended to a canonical ordering of (G, v_1). We say that P_k is a *feasible extension* of Π if $P_0, \ldots, P_k$ or $P_k, \ldots, P_s$, respectively, can also be extended to a canonical ordering of (G, v_1).

Let $G_k = G[V \setminus (P_{k+1}, \ldots, P_s)]$. Kant [28] proved that the path P_k is a feasible extension for $P_{k+1}, \ldots, P_s$ if and only if the following is true:

1. All vertices of P_k are adjacent to some vertex in $P_{k+1}, \ldots, P_s$.
2. If P_k is a chain, then all vertices of P_k have degree 2 in G_k.
3. $G_{k-1} = G[V \setminus (P_k \cup \cdots \cup P_s)]$ is biconnected.

An inner face of G_k is called a *separation face* if its incidence with the outer face of G_k is not only a single path.

Remark 2 *The following two conditions are equivalent for a path P_k on the outer face of G_k, $k \geq 2$:*

1. *$G[V \setminus (P_k \cup \cdots \cup P_s)]$ is biconnected.*
2. *(a) P_k is not incident to a separation face and*
   (b) P_k is a singleton with degree greater than 2 or P_k is a maximal sequence of vertices of degree 2 on the outer face of G_k.

Definition 6 (upper rightist canonical ordering) *A canonical ordering $P_0, \ldots, P_s$ of (G, v_1) is called* upper rightist *if for $k = s-1, \ldots, 0$ the following is true. Let $P \neq P_k$ be a feasible extension of $P_{k+1}, \ldots, P_s$. Then, P is between v_1 and P_k on the clockwise outer facial cycle C_k around G_k.*

Theorem 3 *The upper rightist canonical ordering of a triconnected plane graph with a fixed vertex on the outer face equals its leftist canonical ordering.*

Proof: Let $\Pi = (P_0, \ldots, P_s)$ be the upper rightist canonical ordering and let $\Pi' = (P_0, \ldots, P_{i-1}, P'_i, \ldots, P'_{s'-1}, P'_{s'})$ be the leftist canonical ordering of (G, v_1). For $k = 0, \ldots, s$ let again $G_k = G[V \setminus (P_{k+1} \cup \cdots \cup P_s)] = G[P_0 \cup \cdots \cup P_k]$.

Assume that $P_i \neq P'_i$. Then both, P_i and P'_i are feasible extensions of $P_0, \ldots, P_{i-1}$ and P'_i is to the left of P_i on C_{i-1}. Hence, it is not possible that one of the two is contained in the other. Let $v \in P_i \setminus P'_i$ and $v' \in P'_i \setminus P_i$. Since P'_i is feasible, it follows that $G[V \setminus (P_0 \cup \cdots \cup P_{i-1} \cup P'_i)]$ is connected. Thus, there is a path Q from v to v_n that contains no vertices of P'_i or G_{i-1}.

Let $i < j < s$ be such that $v' \in P_j$. Let w' be the first vertex of Q not in G_j and let w be the vertex that is immediately before w' in Q. Then, w is to the right of v' on the outer facial cycle C_j of G_j. Further, w is adjacent to the vertex w' in $P_{j+1} \cup \cdots \cup P_s$.

Assume first that w is not incident to a separation face. If w has degree 2 in G_j, let P be a maximal sequence of vertices of degree 2 in C_j containing w. Otherwise, let $P = \langle w \rangle$. In both cases P is a feasible extension of $P_{j+1}, \ldots, P_s$ on the right of P_j.

Assume now that w is incident to a separation face f. Consider the separator S of G_j consisting of all the vertices that are incident to both, f and the outer face and consider the components of G_j with respect to S. Then, there is one component that contains G_{i-1} and v'. All other components have to be eliminated before w can become part of a feasible extension (see Figure 4). Hence, all these components have to contain a feasible extension. But all these components are to the right of v'. Thus, P_j was not the rightmost feasible extension. □

Let $G^* = (V^*, E^*)$ be the dual graph of G. Let v_1^* be the dual vertex of the outer face of G and let v_1 be on the outer face of G^*. Kant [27] mentioned that a leftmost canonical ordering of (G, v_1) induces a leftmost canonical ordering on (G^*, v_1^*). We rephrase Kant's construction and show that the result can be extended to the leftist canonical ordering. Note that a similar result holds for *st*-orderings [35].

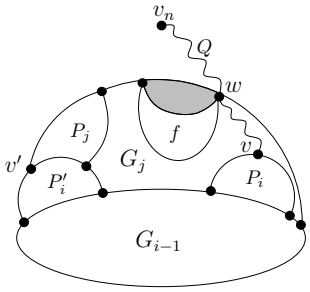


Figure 4: Illustration of the proof of Theorem 3. The gray component has to be eliminated before w can become part of a feasible extension.

Let $\Pi = (P_0, \ldots, P_s)$ be a canonical ordering of (G, v_1). Let E_i be the set of edges of $G_i = G[P_0 \cup \cdots \cup P_i]$, $i = 0, \ldots, s$, and let $E(P_0) = E_0$, $E(P_i) = E_i \setminus E_{i-1}$, $i = 1, \ldots, s$. I. e., $E(P_i)$ consists of all edges that are incident to two vertices in P_i and all cut edges of G_{i-1} that are incident to a vertex of P_i.

Analogously, if $\Pi^* = (P_0^*, \ldots, P_s^*)$ is a partition of the set V^* of faces of G, let E_i^* be the set of edges of $G_i^* = G^*[P_0^* \cup \cdots \cup P_i^*]$, $i = 0, \ldots, s$, and let $E^*(P_0) = E_0^*$, $E^*(P_i^*) = E_i^* \setminus E_{i-1}^*$, $i = 1, \ldots, s$. For $E' \subset E$ let E'_* be the set of dual edges of E'. Further, let v_2^* be the neighbor of v_1^* on the outer facial cycle of G^* in counter clockwise direction.

Definition 7 (dual canonical ordering) *A partition $\Pi^* = (P_0^*, \ldots, P_s^*)$ of V^* into paths is the* dual canonical ordering *of a canonical ordering $\Pi = (P_0, \ldots, P_s)$ of (G, v_1) if and only if $P_0^* = \langle v_1^*, v_2^* \rangle$ and*

$$\begin{aligned} E^*(P_0^*) \cup E^*(P_1^*) &= E(P_s)_* \\ E^*(P_k^*) &= E(P_{s-k+1})_*, \qquad k = 2, \ldots, s-1 \\ E^*(P_s^*) &= E(P_1)_* \cup E(P_0)_*. \end{aligned}$$

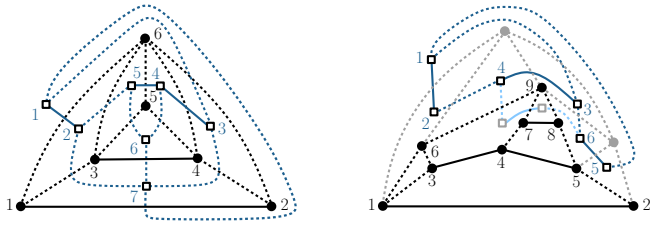
Theorem 4 *Let Π be a canonical ordering of (G, v_1).*

1. *A dual canonical ordering Π^* of Π exists and is uniquely determined. It is a canonical ordering of (G^*, v_1^*).*
2. *Π is the leftist canonical ordering of (G, v_1) if and only if Π^* is the leftist canonical ordering of (G^*, v_1^*).*

Proof: Let $\Pi = (P_0, \ldots, P_s)$.

1. Let v_n^* be the face of G bounded by P_0 and P_1. Then $P_s^* = \langle v_n^* \rangle$. By the definition of the dual canonical ordering, $P_0^* = \langle v_1^*, v_2^* \rangle$. Further, $\langle v_2^*, v_1^*, v_n^* \rangle$ is a path on the outer face of G^* in clockwise direction.

   Again by the definition of a dual canonical ordering, it follows that the subgraph induced by P_0^* and P_1^* is the simple cycle bounding the face of



(a) (b)

Figure 5: (a) A graph G with the leftist canonical ordering Π with $P_0 = \langle 1, 2\rangle$ and $P_s = \langle 6\rangle$, and the dual graph G^* with the leftist canonical ordering Π^* with $P_0^* = \langle 1, 2\rangle$ and $P_s^* = \langle 7\rangle$. Black solid paths are chains in Π, blue solid paths are chains in Π^*. (b) Illustration of the proof of Theorem 4.

G^* in which v_n is located. Hence, Conditions 1 and 3 of Definition 1 are fulfilled for $k = 1$. Condition 2 is fulfilled by the triconnectivity of G^*.

Let C_k^*, $k = 1, \ldots, s$, be the boundary of the outer face of $G_k^* = G^*[P_0^* \cup \cdots \cup P_k^*]$. We will prove the following observation by induction on k while proving Theorem 4(1). The remark is certainly true for $k = 1$.

Remark 3 *Let $k = 1, \ldots, s$. The edges of the simple cycle C_k^* are the duals of the cut edges of G_{s-k} and it holds that the vertices of $P_s \cup \cdots \cup P_{s-k+1}$ are inside the cycle C_k^* and the vertices of G_{s-k} are outside C_k^*.*

Let now $k = 2, \ldots, s-1$ and let $w_1^*, \ldots, w_t^*$ be the faces bounded by P_{s-k+1} and C_{s-k} in the order in which they occur around P_{s-k+1}. Then $P_k^* = \langle w_1^*, \ldots, w_t^*\rangle$. Since each vertex in P_{s-k+1} is adjacent to at least one vertex in $P_{s-k+2} \cup \cdots \cup P_s$, it follows that C_k^* is a simple cycle. Since each of these faces is incident to at least one edge in C_{s-k}, it follows that w_i^*, $i = 1, \ldots, t$, is adjacent to at least one vertex in $P_{k+1}^* \cup \cdots \cup P_s^*$. Assume now that P_k^* is a chain, i. e., that P_{s-k+1} is a singleton $\langle v\rangle$ with more than two neighbors in G_{s-k}. See Figure 5(b) for an illustration. Since C_{s-k} is a simple cycle it follows that all vertices in P_k^* have degree 2 in G_k^*. Finally, the vertices of G inside C_k^* are exactly those that had been inside C_{k-1}^* plus the vertices in P_{s-k+1}. Hence, Remark 3 is true for k.

2. It follows from the construction in Theorem 4(1) that Π^* is the upper rightist canonical ordering of (G^*, v_1^*) and hence, with Lemma 3 that Π^* is the leftist canonical ordering of (G^*, v_1^*).

□

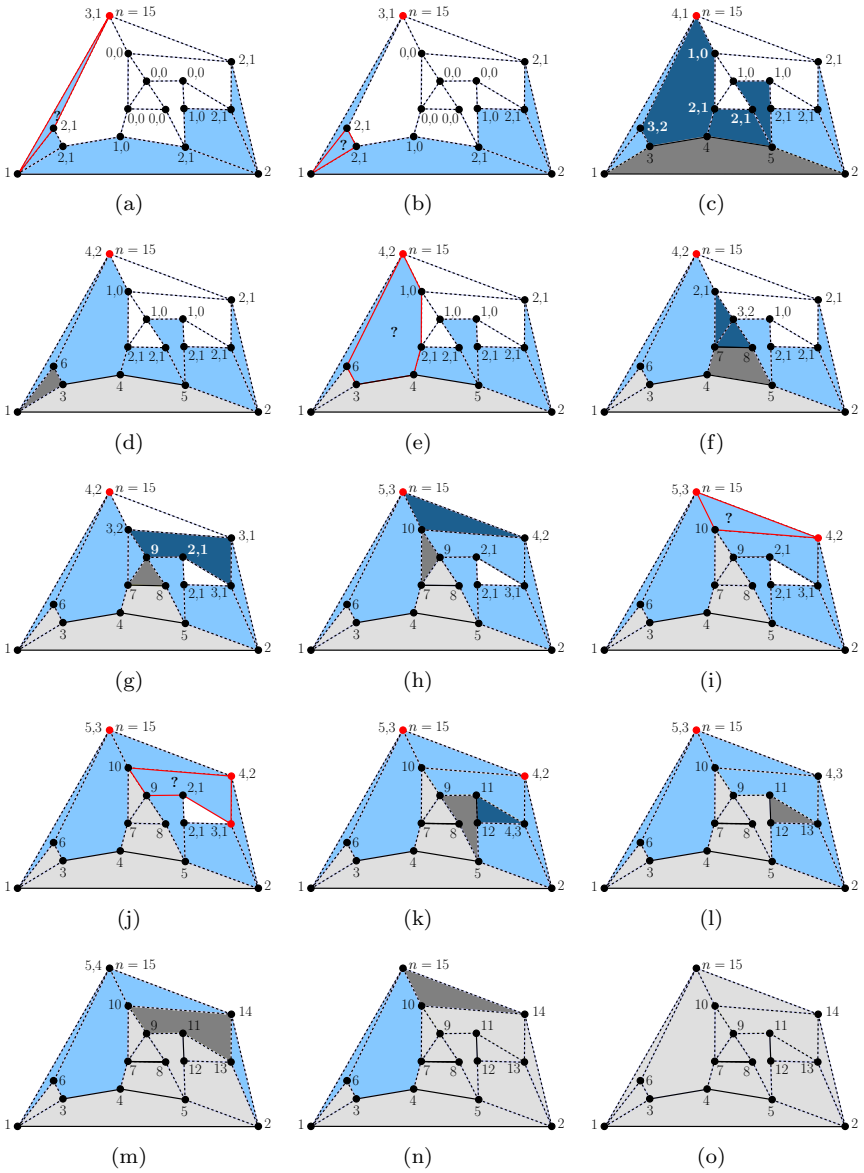


Figure 6: Illustration of Algorithm 2: The light blue faces are the cut faces of G_k, G_k is light gray. The leftmost feasible candidate is dark gray. Algorithm 4 substitutes the dark gray face by the dark blue faces, i. e., by the EXTENSION found by Algorithm 5. Black paths are chains, red vertices are forbidden. If a vertex v is labeled with a pair of numbers, the first number indicates CUTFACES(v) and the second number indicates CUTEDGES(v).

6 Schnyder Woods

In this section, we discuss how the concept of leftist canonical orderings is related to Schnyder woods. Let $G = (V, E)$ be again a planar and triconnected graph and $\langle v_2, v_1, v_n \rangle$ a path on the outer face of G in clockwise direction and let G^* be the dual graph of G.

6.1 Schnyder Woods and Canonical Orderings

Definition 8 (closure) *The* closure of (G, v_1) *is the graph G_∞ that is obtained from G by adding a new vertex v_∞ to the outer face of G and the edges $\{v_1, v_\infty\}$, $\{v_2, v_\infty\}$, and $\{v_n, v_\infty\}$.*

For simplicity, we sometimes denote $v_1 = a_1$, $v_2 = a_2$, $v_n = a_3$ and assume on the labels $i = 1, 2, 3$ a cyclic structure so that $3 + 1 = 1$ and $1 - 1 = 3$.

Definition 9 (Schnyder wood) *A* Schnyder wood of (G, v_1) *is an orientation and labeling of the edges of the closure G_∞ of (G, v_1) with labels 1, 2, and 3 such that:*

1. *Every edge is oriented by one or two opposing directions. If an edge is bioriented, then the two directions have distinct labels.*
2. *The three edges $\{a_i, v_\infty\}$ are oriented towards v_∞ and labeled i, $i = 1, 2, 3$.*
3. *Every vertex $v \neq v_\infty$ has outdegree one in each label. The labels i of the three outgoing edges e_i, $i = 1, 2, 3$, of a vertex v occur in counterclockwise order. Each incoming edge of v with label i enters v in the clockwise sector from e_{i-1} to e_{i+1}. See Figure 7.*
4. *There is no interior face whose boundary is a directed cycle in one label.*

See Figure 8(a) for an example. A Schnyder wood consists of three trees. More precisely, let $i = 1, 2, 3$. Let T_i denote the oriented subgraph of G induced by the edges having label i.

Lemma 5

1. *The graph T_i, $i = 1, 2, 3$, is a spanning tree of G with the unique sink a_i [**?**, Fact 2].*
2. *The graph $T_1^{-1} \cup T_2^{-1} \cup T_3$ in which the orientation of all edges with label 1 or 2 is reversed does not contain any directed cycle of length greater than two [**?**, Lemma 2].*

Let $v \in V$. By the previous lemma, we know that there are oriented paths $T_i(v)$, $i = 1, 2, 3$, from v to a_i, in which all edges are labeled i. The following lemma states that any two of these paths are internally disjoint.

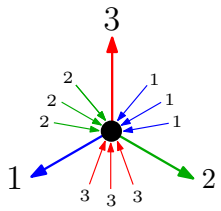


Figure 7: Edge orientations and labels at a vertex.

Lemma 6 ([15, Lemma 6]) *For each $v \in V$ the paths $T_1(v)$, $T_2(v)$, and $T_3(v)$ have only vertex v in common.*

Schnyder woods are closely related to α_0-orientations [**?**] which we define next.

Definition 10 (primal dual superimposition) *The* primal dual superimposition $G^\times = (V^\times, E^\times)$ *of* (G, v_1) *is constructed as follows:*

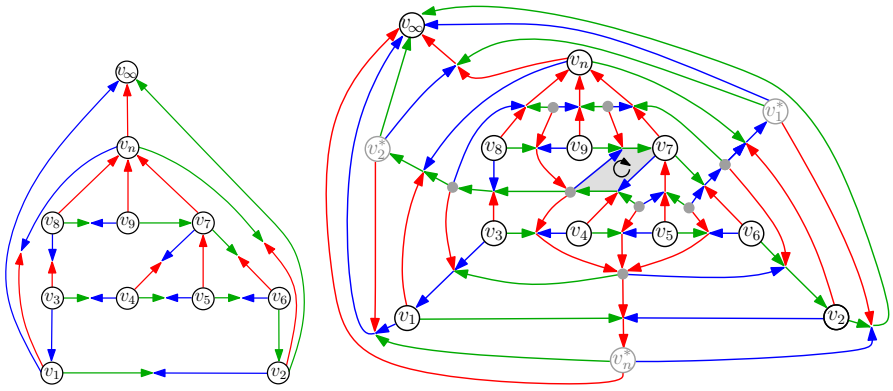
1. *Superimpose the closure G_∞ of (G, v_1) with its dual G^*_∞ such that exactly every edge of G_∞ crosses with its dual edge of G^*_∞ and insert edge-vertices at those crossings.*
2. *Let v_1^* be the face with boundary v_2, v_n, v_∞, let v_2^* be the face with boundary v_1, v_n, v_∞, and let v_n^* be the face with boundary v_1, v_2, v_∞. Add edges $\{v_1^*, v_\infty\}$, $\{v_2^*, v_\infty\}$, and $\{v_n^*, v_\infty\}$.*

Definition 11 (α_0-orientation) *Let $\alpha_0 : V^\times \to \mathbb{N}$ be a function such that $\alpha_0(v) = 3$ for all primal and dual vertices v, $\alpha_0(v_e) = 1$ for all edge-vertices v_e, and $\alpha_0(v_\infty) = 0$. An orientation of the elements of $E^\times$ is called* α_0-orientation *if each vertex $v \in V^\times$ has exactly $\alpha_0(v)$ outgoing edges.*

Felsner [**?**] shows that the Schnyder woods of (G, v_1) are in bijection with the Schnyder woods of the dual graph (G^*_∞, v_1^*), where v_1^* is the dual vertex of the face bounded by v_2, v_n, v_∞, and with the α_0-orientations of $G^\times$ (see Figure 8(b)). However, given only the orientations of a Schnyder wood of (G, v_1), the labels cannot be reconstructed from the underlying orientation [**?**].

There exists a unique α_0-orientation without clockwise cycles of the primal dual superimposition $G^\times$. This is the *minimal α_0-orientation*, where the term minimal refers to the fact that the set of all α_0-orientations of $G^\times$ forms a distributive lattice [**?**]. An α_0-orientation can be made minimal by iterative application of reversing clockwise cycles. A *minimal Schnyder wood* is a Schnyder wood that is associated with the minimal α_0-orientation.

Di Battista et al. [15] construct a Schnyder wood from a canonical ordering. While the minimal Schnyder wood of a triangulated graph is the one associated with the leftist canonical ordering [6], this observation does not hold for triconnected graphs any more. See Figure 8. Moreover, the minimal Schnyder wood cannot always be reconstructed from a canonical ordering. See again Figure 8. The graph has one canonical ordering and at least two different Schnyder woods.



(a) Schnyder wood of (G, v_1) (b) Primal dual superimposition $G^\times$ of (G, v_1)

Figure 8: Schnyder wood associated with the leftist canonical ordering (blue, green, red indicate labels 1, 2, 3, respectively); the corresponding α_0-orientation contains a clockwise cycle.

6.2 Schnyder Woods and Path Partitions

Canonical orderings do not seem to be the right concept to construct a bijection to Schnyder woods since a graph can have more Schnyder woods than canonical orderings. Therefore, we generalize the definition of a canonical ordering to an ordered path partition and show that certain equivalence classes of ordered path partitions are in bijection with Schnyder woods. Further, we show that the leftist ordered path partition corresponds to the minimal Schnyder wood.

Definition 12 (ordered path partition) *Let $P_0 = \langle v_1, v_2 \rangle$, let $P_s = \langle v_n \rangle$, and let V_k and C_k be defined as in Definition 1. An ordered partition $\Pi = (P_0, \ldots, P_s)$ of V into paths is called* ordered path partition *of (G, v_1) if for each $k = 1, \ldots, s-1$:*

1. *C_k is a simple cycle.*
2. *Each vertex in P_k has a neighbor in $V \setminus V_k$.*
3. *Each vertex on C_k has at most one neighbor on P_{k+1}.*

An ordered path partition $\Pi = (P_0, \ldots, P_s)$ of (G, v_1) induces an *orientation* on the edges of G that are not in the paths $P_0, \ldots, P_s$. More precisely, let $e = \{u_1, u_2\}$ be an edge of G such that there are $k_1 < k_2$ with $u_i \in P_{k_i}$, $i = 1, 2$. Then, e is an *outgoing edge* of u_1 and an *incoming edge* of u_2.

Definition 13 (equivalence of ordered path partitions) *Two ordered path partitions $\Pi = (P_0, \ldots, P_s)$ and $\Pi' = (P'_0, \ldots, P'_s)$ are* equivalent *if and only if $\{P_0, \ldots, P_s\} = \{P'_0, \ldots, P'_s\}$ as well as Π and Π' induce the same orientation on the edges.*

Theorem 5 *There is a bijection between the equivalence classes of the ordered path partitions of (G, v_1) and the Schnyder woods of (G, v_1).*

Proof: We show how to map equivalence classes of ordered path partitions to Schnyder woods and Schnyder woods to equivalence classes of ordered path partitions such that the composition of the two maps yields the identity.

From ordered path partitions to Schnyder woods. We extend the construction of Di Battista et al. [15]. Let $\Pi = (P_0, \ldots, P_s)$ be an ordered path partition. Let (u, v) denote the directed edge from u to v and let $\texttt{label}(u, v) = i$ indicate that (u, v) has label i. Orient the three edges $\texttt{label}(a_i, v_\infty) = i$, $i = 1, 2, 3$, and bi-orient $\texttt{label}(a_1, a_2) = 2$ and $\texttt{label}(a_2, a_1) = 1$. Let $k = 1, \ldots, s$ and let $P_k = \langle z_1, \ldots, z_p \rangle$ have left neighbor c_ℓ and right neighbor c_r. We label and orient the edges of P_k and between P_k and C_{k-1} as follows (see Figure 9):

1. $\texttt{label}(z_1, c_\ell) = 1$
2. $\texttt{label}(z_p, c_r) = 2$
3. $\texttt{label}(z_i, z_{i+1}) = 2$ and $\texttt{label}(z_{i+1}, z_i) = 1$, $i = 1, \ldots, p-1$
4. For all $c \in C_{k-1}$ that are incident to a vertex $z \in P_k$ but not to a vertex in $V \setminus V_k$ set $\texttt{label}(c, z) = 3$.

Construction yields a Schnyder wood. If the end vertices of an edge e are in the paths P_i and P_j, $i \leq j$, then e is oriented and labeled in step j. Therefore, Definition 9(1) is true since every edge is oriented exactly once with one or two opposing directions by construction. Definition 9(2) is obviously true.

Since every vertex is contained in exactly one path, every vertex has outdegree one in label 1 and 2. Let z be a vertex of P_k, $k < s$, and let k' be maximal such that z has a neighbor z' in $P_{k'}$. Then, $k' > k$. Therefore, z has an outgoing edge to z' with label 3. Since z has at most one neighbor in $P_{k'}$, this is the only outgoing edge of z with label 3. Also, since $k' > k$, the edge (z, z') appears in the adjacency list of z in the clockwise sector between the outgoing edge with label 1 and the outgoing edge with label 2. By construction, the incoming edges with label 3 appear in the clockwise sector of the outgoing edges with label 2 and 1.

Assume that z has an incoming edge $e = (\hat{z}, z)$ with label 1 that appears in the clockwise sector between the outgoing edges with label 1 and 3. Since $\texttt{label}(\hat{z}, z) = 1$, it follows that $\hat{z}$ is the leftmost vertex of a path $P_{\hat{k}}$ with $\hat{k} > k$ and that z is the left neighbor of $P_{\hat{k}}$. Since $C_{\hat{k}}$ is a simple cycle and by the planarity of G the assumption implies that the outgoing edge of z with label 3 is an incoming edge of $P_{\hat{k}}$ to the right of $\hat{z}$. By Definition 12, z has at most one vertex on $P_{\hat{k}}$. This is a contradiction. See Figure 9. The same argumentation holds for the incoming edges with label 2. This completes Definition 9(3).

We show by induction that there is no cycle in one label (Definition 9(4)). This is for sure true for the first path $P_0 = \langle v_1, v_2 \rangle$. Assume that in G_{k-1} there is no cycle in one label and that P_k is the next path. When adding P_k, there does not exist a directed path in any label between two vertices on C_{k-1} using vertices of P_k.

Independence of representatives. All representatives have the same paths. Also, two ordered path partitions are only equivalent if they induce the same orientation and, therefore, induce the same Schnyder wood.

From Schnyder woods to ordered path partitions. Let a Schnyder wood of (G, v_1) be given. A path $P = \langle z_1, \ldots, z_p \rangle$ of G is a *1-2 labeled path* if the edges $\{z_i, z_{i+1}\}$, $i = 1, \ldots, p-1$, are oriented from z_i to z_{i+1} with label 2, and from z_{i+1} to z_i with label 1. We define a partial ordering $\prec$ on the partition of V into maximal 1-2 labeled paths. Let e be an edge between two maximal 1-2 labeled paths P and P'. Then $P \prec P'$ if

1. e is oriented from P to P' and labeled 3 or
2. e is oriented from P' to P and labeled 1 or 2.

We will show that the transitive closure of these two conditions indeed yields a partial ordering. We will then show that the set of all linear extensions $\Pi = (P_0, \ldots, P_s)$ of this partial ordering $\prec$ defines an equivalence class of ordered path partitions. This will be the image of the given Schnyder wood.

$\prec$ is acyclic Assume that there is a sequence $Q_0 \prec \cdots \prec Q_k = Q_0$ of ascending maximal 1-2 colored paths such that the first and the last element is the same. Then there is a cycle $C = \langle z_0, \ldots, z_p \rangle, p > 2$, in G such that the edges $\{z_i, z_{i+1}\}$, $i = 0, \ldots, p-1$, are oriented from z_i to z_{i+1} and labeled 3 or oriented from z_{i+1} to z_i and labeled 1 or 2. Note that some edges of C may also be bioriented. Especially, C may contain edges of the 1-2 labeled paths. However, this contradicts Lemma 5(2).

$\Pi = (\mathbf{P_0}, \ldots, \mathbf{P_s})$ is an ordered path partition. $\langle v_1, v_2 \rangle$ is a maximal 1-2 labeled path. From each vertex there is an oriented path labeled 1 to v_1. Hence, $P_0 = \langle v_1, v_2 \rangle$. All edges incident to v_n are labeled 3. Hence, $\langle v_n \rangle$ is a maximal 1-2 labeled path. From each vertex there is an oriented path labeled 3 to v_n. Hence, $P_s = \langle v_n \rangle$.

Next, we prove by induction that C_k, $k = 1, \ldots, s-1$, is a simple cycle and that C_k, $k = 0$, consists of a single edge. This claim is true for $k = 0$. Let now $k > 0$, let $i = 1, 2$, and let c_i be the vertex not in P_k that is incident to the edge labeled i and oriented away from P_k. By the definition of $\prec$, we have $c_1, c_2 \in V_{k-1}$. Again by the definition of $\prec$, the paths $T_3(u), u$ on P_k may not intersect C_{k-1}. Hence, P_k is contained in the exterior of C_{k-1} and, thus, c_1, c_2 are on C_{k-1}. By Lemma 6, it follows that $c_1 \neq c_2$. Hence, P_k has at least two neighbors on C_{k-1}. Thus, C_k is a simple cycle.

Let now $k = 1, \ldots, s-1$ and let u be a vertex of P_k. Then, there is an edge labeled 3 and oriented away from u. This edge is incident to a vertex of $V \setminus V_k$.

From the definition of Schnyder woods and of $\prec$ it follows that among the edges between C_{k-1} and P_k there are exactly two that are oriented towards C_{k-1} – one with label 1 and one with label 2. All other edges are labeled 3 and oriented towards P_k. Assume now that there is a vertex c on C_{k-1} that has two neighbors u, u' in P_k. The edges $\{c, u\}$, $\{c, u'\}$ cannot both be labeled 3. Otherwise, c had two outgoing edges with the same label. By Lemma 6, $\{c, u\}$, $\{c, u'\}$ cannot be labeled 1 and 2, respectively. So assume that $\{c, u\}$ is labeled 1 and that $\{c, u'\}$ is labeled 3. Consider the cycle C bounded by $\{c, u\}$, $\{c, u'\}$ and a part of P_k. By Definition 9(3) there have to be vertices x_i, $i = 1, 2$, in the interior of C such that the edges $\{c, x_i\}$ are oriented towards x_i and labeled i. Now, we have on one hand $x_i \in V_{k-1}$. On the other hand, the simple cycle C_{k-1} cannot bound a region that contains c, x_1, x_2, v_1, and v_2, but does not intersect P_k. The case in which $\{c, u\}$ is labeled 2 is symmetric.

It is easy to see that the two constructions are inverse to each other. □

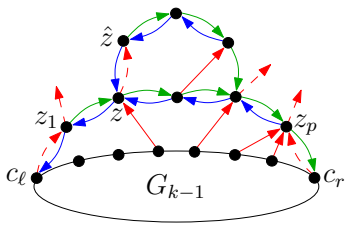


Figure 9: Construction of a Schnyder wood from an ordered path partition.

Definition 14 (leftist ordered path partition) *An ordered path partition $P_0, \ldots, P_s$ is called* leftist *if for $k = 0, \ldots, s-1$ and an ordered path partition $P'_0, \ldots, P'_{s'}$ with $P_i = P'_i$, $i = 0, \ldots, k$, the following is true. Let c_ℓ (c_r) be the left (right) neighbor of P_{k+1} and $c_{\ell'}$ ($c_{r'}$) be the left (right) neighbor of P'_{k+1}. Then, $\ell \leq \ell'$ and $r \leq r'$.*

Note that ℓ and r in the above definition can be simultaneously minimized. Assume that P'_{k+1} is contained in P_{k+1} with $\ell < \ell'$ and $r' < r$. Let z_1 be the neighbor of c_ℓ in P_{k+1} and z_p the neighbor of $c_{r'}$ in P_{k+1}. Then, the subpath $P = \langle z_1, \ldots, z_p \rangle$ of P_{k+1} fulfills the properties of Definition 12, has the same left neighbor as P_{k+1}, and a right neighbor that is to the left of c_r. Moreover, analogously to canonical orderings, a sequence $P_0, \ldots, P_i$ of paths can be extended to an ordered path partition if and only if $P_0, \ldots, P_i$ fulfill the properties of Definition 12 and $G[V \setminus (P_0 \cup \cdots \cup P_i)]$ is connected. The latter is

fulfilled for P since P is a subpath of P_{k+1}, $G[V \setminus (P_0 \cup \cdots \cup P_{k+1})]$ is connected, and each vertex of P_{k+1} is adjacent to a vertex of $G[V \setminus (P_0 \cup \cdots \cup P_{k+1})]$.

We now show that the leftist ordered path partition corresponds to the minimal Schnyder wood. The algorithm of Fusy et al. [**?**] to compute the minimal element of an α_0-orientation reuses the idea of the algorithm of Kant [28]. Let G_k and C_k be as in Definition 1. A vertex v on C_k is *eligible* if

1. it is incident to at least one edge in $G[V \setminus V_k]$ and
2. it is not incident to a separation face.

Fusy et al. [**?**] iteratively eliminate the rightmost eligible vertex and its incident faces from the outer face of $G^\times$ until the graph is shrunk to the edge $\{v_1, v_2\}$. The vertices v_1 and v_2 are considered to be blocked until only the edge $\{v_1, v_2\}$ is left. Let v be the rightmost eligible vertex that is eliminated in step $s - k + 1$. Let P_k be the path that consists of v and a maximal chain of vertices with degree two on C_k to the left of v. Then, P_k is the next path that is eliminated and $\Pi = (P_0, \ldots, P_s)$ is the corresponding ordered path partition. It follows that Π is the upper rightist ordered path partition in the following sense:

Definition 15 (upper rightist ordered path partition) *An ordered path partition $P_0, \ldots, P_s$ of (G, v_1) is called* upper rightist *if for $k = 1, \ldots, s$ and an ordered path partition $P'_0, \ldots, P'_{s'}$ with $P_{s-i} = P'_{s'-i}$, $i = 0, \ldots, k-1$, the following is true. Let $P_{s-k} = \langle z_1, \ldots, z_p \rangle$ and let $P'_{s'-k} = \langle z'_1, \ldots, z'_{p'} \rangle$, then z'_1 is between v_1 and z_1, and $z'_{p'}$ is between v_1 and z_p on the clockwise outer facial cycle C_{s-k} around G_{s-k}.*

Theorem 6 *The leftist ordered path partition corresponds to the minimal Schnyder wood.*

Proof: It remains to show that the upper rightist ordered path partition equals the leftist ordered path partition. Let $\Pi = (P_0, \ldots, P_s)$ be the upper rightist ordered path partition, let $\Pi' = (P_0, \ldots, P_{i-1}, P'_i, \ldots, P'_{s'-1}, P'_{s'})$ be the leftist ordered path partition of (G, v_1), and assume $P_i \neq P'_i$.

If P_i is contained in P'_i or P'_i is contained in P_i, then P'_i does not fulfill the requirements of a leftist ordered path partition. Otherwise, the argumentation is the same as in Theorem 3. □

We adapt Algorithm 2 such that it computes the leftist ordered path partition in the following way. Replace line ▶^{1} in Algorithm 3 by:

while $j > 0$ **and not** *forbidden*(z_j) **do**

Replace line ▶^{2} in the same algorithm by:

if $j = 0$ **then**

Finally, replace the **if**-loop ▼ in Algorithm 4 by:

let $\langle (z_0, z_1), (z_1, z_2), \ldots, (z_p, z_{p+1}) \rangle :=$ CANDIDATE.CHAIN
if *singular*(z_p) **then**
⌊ remove neighboring items with z_p as singleton from BELT

Theorem 7 *The modification of Algorithm 2 as specified above computes the leftist ordered path partition of a triconnected planar graph in linear time.*

Proof: The running time of Algorithm 2 is not affected by the changes.

The difference to a canonical ordering is that a path of the leftist ordered path partition can be a chain with an attached singleton to the right. This path would be split into a singleton and a chain in a canonical ordering.

The modifications in lines $\blacktriangleright^1$ and $\blacktriangleright^2$ in Algorithm 3 guarantee that the algorithm does not skip a candidate if it contains a singular vertex. Since we process the candidate faces from left to right and the vertices in a face from right to left, only the rightmost vertex of a path in a leftist ordered path partition can be singular. If a vertex is forbidden, the algorithm goes to the next face.

If the rightmost vertex of a path is singular, all neighboring items are still removed from the belt (**if**-loop $\blacktriangledown$ in Algorithm 4). □

Acknowledgment

The authors want to thank Michael Baur for useful discussion, helpful hints for the pseudocodes, and the implementation of the algorithm. We also thank the anonymous referee who prompted us to consider Schnyder woods.

References

[1] J. Barbay, L. C. Aleardi, M. He, and I. Munro. Succinct Representation of Labeled Graphs. In T. Tokuyama, editor, *Proceedings of the 18th International Symposium on Algorithms and Computation (ISAAC'07)*, volume 4835 of *Lecture Notes in Computer Science*, pages 316–328. Springer, 2007.

[2] G. Barequet, M. T. Goodrich, and C. Riley. Drawing Planar Graphs with Large Vertices and Thick Edges. *Journal of Graph Algorithms and Applications*, 8(1):3–20, 2004.

[3] T. C. Biedl. Drawing Planar Partitions I: LL-Drawings and LH-Drawings. In *Proceedings of the 14th Annual Symposium on Computational Geometry*, pages 287–296. ACM Press, 1998.

[4] T. C. Biedl and M. Kaufmann. Area-Efficient Static and Incremental Graph Drawings. In G. J. Woeginger, editor, *Proceedings of the 5th Annual European Symposium on Algorithms (ESA'97)*, volume 1284 of *Lecture Notes in Computer Science*, pages 37–52. Springer, 1997.

[5] P. Bose, J. Gudmundsson, and M. Smid. Constructing Plane Spanners of Bounded Degree and Low Weight. *Algorithmica*, 42(3–4):249–264, 2005.

[6] E. Brehm. 3-Orientations and Schnyder 3-Tree-Decompositions. Master's thesis, FU Berlin, 2000.

[7] Y.-T. Chiang, C.-C. Lin, and H.-I. Lu. Orderly Spanning Trees with Applications to Graph Encoding and Graph Drawing. In *Proceedings of the 12th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'01)*, pages 506–515, 2001.

[8] M. Chrobak and G. Kant. Convex Grid Drawings of 3-Connected Planar Graphs. *International Journal of Computational Geometry and Applications*, 7(3):211–223, 1997.

[9] M. Chrobak and S.-I. Nakano. Minimum-Width Grid Drawings of Plane Graphs. *Computational Geometry*, 11(1):29–54, 1998.

[10] M. Chrobak and T. H. Payne. A Linear-Time Algorithm for Drawing a Planar Graph on a Grid. *Information Processing Letters*, 54(4):241–246, 1995.

[11] R. C.-N. Chuang, A. Garg, X. He, M.-Y. Kao, and H.-I. Lu. Compact Encodings of Planar Graphs via Canonical Orderings and Multiple Parentheses. In K. G. Larsen, S. Skyum, and G. Winskel, editors, *Proceedings of the 25th International Colloquium on Automata, Languages and Programming (ICALP'98)*, volume 1443 of *Lecture Notes in Computer Science*, pages 118–129. Springer, 1998.

[12] H. de Fraysseix and P. O. Mendez. Regular Orientations, Arboricity, and Augmentation. In *DIAMCS International Workshop*, volume 894 of *Lecture Notes in Computer Science*, pages 111–118. Springer, 1995.

[13] H. de Fraysseix, J. Pach, and R. Pollack. Small Sets Supporting Fáry Embeddings of Planar Graphs. In *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing (STOC'88)*, pages 426–433. ACM Press, 1988.

[14] H. de Fraysseix, J. Pach, and R. Pollack. How to Draw a Planar Graph on a Grid. *Combinatorica*, 10(1):41–51, 1990.

[15] G. Di Battista, R. Tamassia, and L. Vismara. Output-Sensitive Reporting of Disjoint Paths. *Algorithmica*, 23(4):302–340, 1999.

[16] E. Di Giacomo, W. Didimo, and G. Liotta. Radial Drawings of Graphs: Geometric Constraints and Trade-Offs. *Journal of Discrete Algorithms*, 6(1):109–124, 2008.

[17] E. Di Giacomo, W. Didimo, G. Liotta, and S. K. Wismath. Drawing Planar Graphs on a Curve. In H. L. Bodlaender, editor, *Proceedings of the 29th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'03)*, volume 2880 of *Lecture Notes in Computer Science*, pages 192–204. Springer, 2003.

[18] E. Di Giacomo, W. Didimo, G. Liotta, and S. K. Wismath. Curve-Constrained Drawings of Planar Graphs. *Computational Geometry*, 30(2):1–23, 2005.

[19] V. Dujmović, M. Suderman, and D. R. Wood. Really Straight Graph Drawings. In *Proceedings of the 12th International Symposium on Graph Drawing (GD'04)*, volume 3383 of *Lecture Notes in Computer Science*, pages 122–132. Springer, 2005.

[20] U. Fößmeier, G. Kant, and M. Kaufmann. 2-Visibility Drawings of Planar Graphs. In *Proceedings of the 4th International Symposium on Graph Drawing (GD'96)*, volume 1090 of *Lecture Notes in Computer Science*, pages 155–168. Springer, 1997.

[21] M. T. Goodrich and C. G. Wagner. A Framework for Drawing Planar Graphs with Curves and Polylines. *Journal of Algorithms*, 37(2):399–421, 2000.

[22] C. Gutwenger and P. Mutzel. Planar Polyline Drawings with Good Angular Resolution. In S. H. Whitesides, editor, *Proceedings of the 6th International Symposium on Graph Drawing (GD'98)*, volume 1547 of *Lecture Notes in Computer Science*, pages 167–182. Springer, 1999.

[23] D. Harel and M. Sardas. An Algorithm for Straight-Line Drawing of Planar Graphs. *Algorithmica*, 20:119–135, 1998.

[24] X. He. On Floor-Plan of Plane Graphs. *SIAM Journal on Computing*, 28(6):2150–2167, 1999.

[25] X. He, M.-Y. Kao, and H.-I. Lu. Linear-Time Succinct Encodings of Planar Graphs via Canonical Orderings. *SIAM Journal on Discrete Mathematics*, 12(3):317–325, 1999.

[26] G. Kant. Drawing Planar Graphs using the lmc-Ordering. In *Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'92)*, pages 101–110. IEEE Computer Society, 1992.

[27] G. Kant. Drawing Planar Graphs using the lmc-Ordering. Technical Report RUU-CS-92-33, Department of Information and Computing Sciences, Utrecht University, 1992.

[28] G. Kant. Drawing Planar Graphs Using the Canonical Ordering. *Algorithmica*, 16(4):4–32, 1996.

[29] G. Kant. A More Compact Visibility Representation. *International Journal of Computational Geometry and Applications*, 7(3):197–210, 1997.

[30] G. Kant and X. He. Regular Edge Labeling of 4-Connected Plane Graphs and its Applications in Graph Drawing Problems. *Theoretical Computer Science*, 172(1-2):175–193, 1997.

[31] C.-C. Lin, H.-I. Lu, and I.-F. Sun. Improved Compact Visibility Representation of Planar Graph via Schnyder's Realizer. *SIAM Journal on Discrete Mathematics*, 18(1):19–29, 2004.

[32] K. Miura, M. Azuma, and T. Nishizeki. Canonical Decomposition, Realizer, Schnyder Labeling and Orderly Spanning Trees of Plane Graphs. *International Journal of Foundations of Computer Science*, 16(1):117–141, 2005.

[33] S.-I. Nakano. Planar Drawings of Plane Graphs. *IEICE Transactions on Information and Systems*, E83-D(3):384–391, 2000.

[34] W. Schnyder. Embedding Planar Graphs on the Grid. In *Proceedings of the 1st Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'90)*, pages 138–148, 1990.

[35] R. Tamassia and I. G. Tollis. A Unified Approach to Visibility Representations of Planar Graphs. *Discrete and Computational Geometry*, 1:321–341, 1986.

[36] K. Wada and W. Chen. Linear Algorithms for a k-Partition Problem of Planar Graphs. In J. Hromkovic and O. Sýkora, editors, *Proceedings of the 24th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'98)*, Lecture Notes in Computer Science, pages 324–336. Springer, 1998.