Journal of Graph Algorithms and Applications
http://jgaa.info/
vol. 7, no. 2, pp. 221–241 (2003)

Upward Embeddings and Orientations of Undirected Planar Graphs

Walter Didimo
Dipartimento di Ingegneria Elettronica e dell'Informazione
Università di Perugia
via G. Duranti 93, 06125 Perugia, Italy.
http://www.diei.unipg.it/~didimo/
didimo@diei.unipg.it

Maurizio Pizzonia
Dipartimento di Informatica e Automazione
Università di Roma Tre
via della Vasca Navale 79, 00146 Roma, Italy.
http://www.dia.uniroma3.it/~pizzonia/
pizzonia@dia.uniroma3.it

Abstract

An *upward embedding* of an embedded planar graph specifies, for each vertex v, which edges are incident on v "above" or "below" and, in turn, induces an *upward orientation* of the edges from bottom to top. In this paper we characterize the set of all upward embeddings and orientations of an embedded planar graph by using a simple flow model, which is related to that described by Bousset [3] to characterize bipolar orientations. We take advantage of such a flow model to compute upward orientations with the minimum number of sources and sinks of 1-connected embedded planar graphs. We finally devise a new algorithm for computing visibility representations of 1-connected planar graphs using our theoretic results.

Communicated by Giuseppe Liotta and Ioannis G. Tollis: submitted October 2001; revised April 2002.

Research partially supported by "Progetto ALINWEB: Algoritmica per Internet e per il Web", MIUR Programmi di Ricerca Scientifica di Rilevante Interesse Nazionale.

1 Introduction

Let G be an undirected planar graph with a given planar embedding. Loosely speaking, an *upward embedding* (also called an *upward representation*) of G is specified by splitting, for each vertex v of G, the ordered circular list of the edges that are incident on v into two linear lists (from left to right) $E_{above}(v)$ and $E_{below}(v)$, in such a way that there exists a planar drawing Γ of G with the following properties: (i) all the edges are monotone in vertical direction; (ii) for each vertex v the edges in $E_{above}(v)$ ($E_{below}(v)$) are incident on v above (below) the horizontal line through v.

A drawing Γ that verifies properties (i) and (ii) is said to be an *upward drawing* of G. An orientation of all edges of Γ from bottom to top defines an orientation of G, which we call an *upward orientation* of G. Hence, each upward embedding of G induces an upward orientation of G. Figure 1 shows an upward embedding of an embedded planar graph and the upward orientation induced by it.

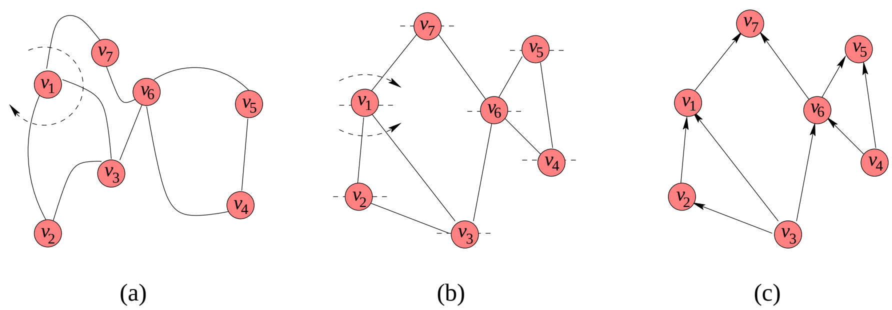


Figure 1: (a) An embedded planar graph. (b) An upward embedding of the embedded planar graph. For each vertex v_i of the graph the edges in $E_{below}(v_i)$ and $E_{above}(v_i)$ are drawn incident below and above the horizontal line through v_i, respectively. (c) The upward orientation induced by the upward embedding.

An embedded planar graph has in general many upward embeddings and upward orientations within the given embedding. Although upward embeddings and orientations have been widely studied within specific theoretic and application domains, as far as we know no complete combinatorial characterizations have been provided in the case of general embedded planar graphs. In the present paper we investigate this problem and we show how our theoretic results have interesting applicability in graph drawing.

An important class of upward orientations, deeply studied in the literature, is represented by the so called *bipolar orientations* (or *st-orientations*). A bipolar orientation of an undirected planar graph G is an upward orientation of G with exactly one source s (vertex without in-edges) and one sink t (vertex without out-edges). A bipolar orientation of G with source s and sink t exists if and only if $G \cup \{(s,t)\}$ is biconnected. Finding a bipolar orientation of a planar graph is the first step of many algorithms in graph theory and graph drawing. A complete and elegant study of the properties of bipolar orientations has been provided by de Fraysseix et. al. [5], and a characterization of bipolar orientations in terms of a network flow model has been described by Bousset [3].

Czyzowicz, Kelly and Rival [14, 13, 4, 16] provide several theoretic results about upward orientations and upward drawings of ordered set and planar lattices, that is, special classes of combinatorial structures.

Several results on upward embeddings of digraphs have been also provided in the literature. In this case, the orientation of the edges of the graph is given, and a classical problem consists of finding an upward (planar) embedding that preserves such an orientation. Clearly, an upward embedding of a digraph might not exist. Bertolazzi et al. [1] describe a polynomial time algorithm for testing the existence of upward embeddings of a digraph within a given planar embedding. The algorithm is also able to construct an upward embedding if there exists one. In the variable embedding setting the upward planarity testing problem is NP-complete [9], but it can be solved in polynomial time for digraphs with a single source [2].

The main contributions of this paper are listed below:

- Starting from the properties on upward planarity of digraphs given in [1], we provide a complete characterization of the set of all upward embeddings and orientations of any embedded planar graph (Section 3.1). It is based on a network flow model, which is a generalization of that used by Bousset [3] for characterizing bipolar orientations. In particular, if the graph is biconnected, our flow model also captures all bipolar orientations of the graph.

- We describe flow based polynomial time algorithms for computing upward embeddings of the input graph. Such algorithms allow us to handle partial specifications of the upward embedding (Section 3.1). Further, we provide a polynomial time algorithm to compute upward orientations with the minimum number of sources and sinks (Section 3.2). Upward orientations with the minimum number of sources and sinks can be viewed as a natural extension of the concept of bipolar orientations to 1-connected graphs.

- We describe a simple technique to compute visibility representations of 1-connected planar graphs (Section 4), which can be of practical interest for graph drawing applications. It is based on the computation of an upward embedding of the graph, and does not require running any augmentation algorithm to initially make the graph biconnected. Compared to a standard technique that uses the good approximation algorithm described by Fialko and Mutzel [8] to make the graph biconnected, the algorithm we propose is theoretically faster, simpler to implement, and achieves similar results in terms of area of the visibility representation.

In Section 2 we recall formal definitions and known results on upward embeddings and orientations of undirected planar graphs.

2 Basic Definitions and Results on Upward Embeddings

A graph is 1-*connected* (or *connected*) if there exists a path between any pair of its vertices. A vertex of the graph whose removal disconnects the graph is called a *cutvertex*. A connected graph is 2-*connected* (or *biconnected*) if it has no cutvertex. Given a 1-connected graph G, a *biconnected component* (or *block*) of G is a maximal biconnected subgraph of G. Observe that each cutvertex of G belongs to at least two distinct blocks of G, and that each edge of G belongs to exactly one block of G. The decomposition of a graph into its blocks can be easily done in linear time [18].

A *drawing* Γ of a graph G maps each vertex u of G into a point p_u of the plane and each edge (u, v) of G into a Jordan curve between p_u and p_v. Γ is *planar* if two distinct edges never intersect except at common end-points. G is *planar* if it admits a planar drawing. A planar drawing Γ of G divides the plane into topologically connected regions called *faces*. Exactly one of these faces is unbounded, and it is said to be *external*; the others are called *internal* faces. Also, for each vertex v of G, Γ induces a circular clockwise ordering of the edges incident on v. The choice ϕ of such an ordering for each vertex of G and of an external face is called a *planar embedding* of G. A planar graph G with a given planar embedding ϕ is called an *embedded planar graph* and denoted by G_ϕ. A *drawing* of G_ϕ is a planar drawing of G that induces ϕ as the planar embedding.

Let G_ϕ be an (undirected) embedded planar graph. An *upward embedding* $\mathcal{E}_\phi$ of G_ϕ is a splitting of the adjacency lists of all vertices of G_ϕ such that:

(E1) For each vertex v of G_ϕ the circular clockwise list $L(v)$ of the edges incident on v is split into two linear lists (from left to right), $E_{below}(v)$ and $E_{above}(v)$, so that the circular list obtained by concatenating $E_{above}(v)$ and the reverse of $E_{below}(v)$ is equal to $L(v)$.

(E2) There exists a planar drawing $\Gamma(\mathcal{E}_\phi)$ of G_ϕ such that all the edges are monotone in vertical direction and for each vertex v of G_ϕ the edges of $E_{below}(v)$ and $E_{above}(v)$ are incident on v below and above the horizontal line through v, respectively. We say that $\Gamma(\mathcal{E}_\phi)$ is a *drawing* of $\mathcal{E}_\phi$ and an *upward drawing* of G_ϕ.

From (E2) the following is immediate.

Property 1 *Given an upward embedding of G_ϕ, for each edge $e = (u, v)$ of G_ϕ either $e \in E_{above}(u) \cap E_{below}(v)$ or $e \in E_{below}(u) \cap E_{above}(v)$.*

An upward embedding $\mathcal{E}_\phi$ of G_ϕ uniquely induces an *upward orientation* $\mathcal{O}_\phi$ of G_ϕ. Namely, for each edge $e = (u, v)$ such that $e \in E_{above}(u)$ and $e \in E_{below}(v)$, we orient e from u to v (see Figure 1). Conversely, an upward orientation defines in general a class of possible upward embeddings inducing it (see Figure 2). A *source* of $\mathcal{E}_\phi$ is a vertex v of G_ϕ such that $E_{below}(v)$ is empty. A source has only out-edges with respect to orientation $\mathcal{O}_\phi$. A *sink* of $\mathcal{E}_\phi$ is a vertex v of G_ϕ such that $E_{above}(v)$ is empty. A sink has only in-edges with respect to $\mathcal{O}_\phi$.

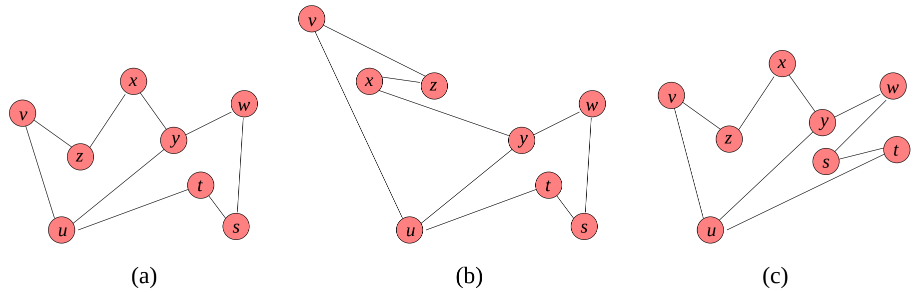


Figure 2: Three different upward embeddings that induce the same upward orientation.

Given a vertex v of G_ϕ, we denote by $deg(v)$ the number of edges incident on v. An *angle* of G_ϕ at vertex v is a pair of clockwise consecutive edges incident on v. In particular, if $deg(v) = 1$, and if we denote by e the edge incident on v, (e, e) is an angle. Given a splitting of the adjacency lists of G_ϕ that verifies (E1), an angle (e_1, e_2) at vertex v of G_ϕ can be of three different types (see Figure 3 for an example):

- *large*: (i) both e_1 and e_2 belong to $E_{below}(v)$ ($E_{above}(v)$), and (ii) e_1 and e_2 are the first (last) edge and the last (first) edge of $E_{below}(v)$ ($E_{above}(v)$), respectively. We associate a label `L` with a large angle.
- *flat*: if: (i) $e_1 \in E_{below}(v)$ and $e_2 \in E_{above}(v)$ or, (ii) $e_1 \in E_{above}(v)$ and $e_2 \in E_{below}(v)$. We associate a label `F` with a flat angle.
- *small*: in all the other cases. We associate a label `S` with a small angle.

Figure 4 shows the labeling of the angles of an embedded planar graph G_ϕ determined by an upward embedding $\mathcal{E}_\phi$. Each drawing of $\mathcal{E}_\phi$ maps the angles of G_ϕ to geometric angles such that large and small angles always correspond to geometric angles larger and smaller than 180 degrees, respectively. Both the two edges that form a large or a small angle at vertex v are incident on v either above or below the horizontal line through v. Instead, a flat angle at vertex v corresponds to a geometric angle that can be either larger or smaller than 180 degrees, but in any case one of its edges is incident on v above the horizontal line through v while the other edge is incident on v below the same line.

Let f be a face of G_ϕ. We call *border* of f the alternating circular list of vertices and edges that form the boundary of f. Note that, if the graph is not biconnected an edge or a vertex may appear more than once in the border of f. We say that an angle (e_1, e_2) at vertex v *belongs* to face f if e_1, e_2, and v belong

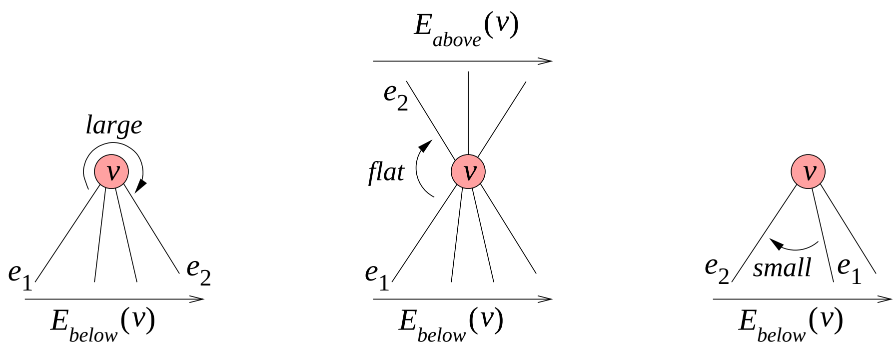


Figure 3: Examples of large, flat, and small angles.

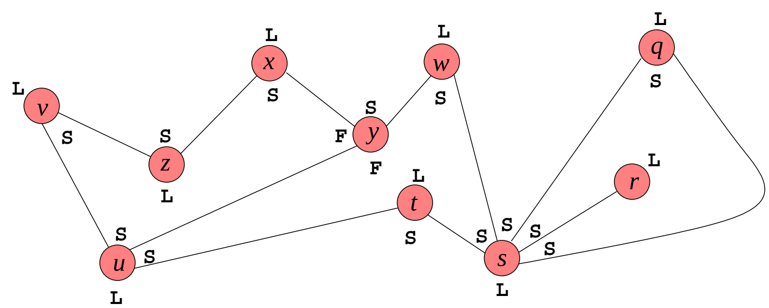


Figure 4: The labeling of the angles of an embedded planar graph determined by an upward embedding of the graph.

to the border of f. The *degree* of f, denoted by $deg(f)$, is the number of edges in the border of f. Observe that, $deg(f)$ is equal to the number of angles of f.

Consider now any labeling of the angles of G_ϕ with labels L, S, and F. For each face f of G_ϕ denote by $L(f)$, $S(f)$, and $F(f)$ the number of angles that belong to f with label L, S, and F, respectively. Also, for each vertex v of G_ϕ denote by $L(v)$, $S(v)$, and $F(v)$ the number of angles at vertex v with label L, S, and F, respectively. The following lemma is a direct consequence of a known result on upward planarity [1].

Lemma 1 *Let $\mathcal{E}_\phi$ be a splitting of the adjacency lists of G_ϕ that verifies* (E1)*, and consider the labeling of the angles of G_ϕ determined by it. $\mathcal{E}_\phi$ is an upward embedding of G_ϕ if and only if the following properties hold:*

(FIN) $S(f) = L(f) + 2$, *for each internal face f of G_ϕ.*

(FEX) $S(f) = L(f) - 2$, *for the external face f of G_ϕ.*

(VL0) $F(v) = 2$, $S(v) = deg(v) - 2$, *and* $L(v) = 0$, *for each vertex* v *of* G_ϕ *such that both* $E_{above}(v)$ *and* $E_{below}(v)$ *are not empty.*

(VL1) $F(v) = 0$, $S(v) = deg(v) - 1$, *and* $L(v) = 1$, *for each vertex* v *of* G_ϕ *such that either* $E_{above}(v)$ *or* $E_{below}(v)$ *is empty.*

Properties (VL0) and (VL1) of Lemma 1 state that if $\mathcal{E}_\phi$ is an upward embedding of G_ϕ, each source or sink of $\mathcal{E}_\phi$ has exactly one large angle and no flat angle, while each vertex that is neither a source nor a sink has exactly two flat angles and no large angle. The next lemma provides a different formulation for properties (FIN) and (FEX).

Lemma 2 *Properties* (FIN) *and* (FEX) *of* Lemma 1 *are equivalent to the following properties:*

(FIN') $deg(f) - 2 = 2L(f) + F(f)$, *for each internal face* f *of* G_ϕ.

(FEX') $deg(f) + 2 = 2L(f) + F(f)$, *for the external face* f *of* G_ϕ.

Proof:
Property (FIN) is equivalent to property (FIN') since $deg(f) = L(f)+S(f)+F(f)$. The equivalence between properties (FEX) and (FEX') can be proved analogously.

q.e.d.

Let G_ϕ be an embedded planar graph, $\mathcal{E}_\phi$ be an upward embedding of G_ϕ, and $\mathcal{O}_\phi$ be the upward orientation induced by $\mathcal{E}_\phi$. Also, denote by D_ϕ the directed graph obtained by G_ϕ orienting its edges according to $\mathcal{O}_\phi$.

In Section 4 we need to compute a super-digraph of D_ϕ with only one source and one sink (*st-digraph*) and preserving the upward embedding $\mathcal{E}_\phi$ when restricted to D_ϕ. In the following we recall an algorithm for this purpose. Further details can be found in [1].

Given a face f of D_ϕ, a vertex v of f with consecutive incident edges e_1 and e_2 on the boundary of f is a *switch* of f if e_1 and e_2 are both incoming or both outgoing v (note that e_1 and e_2 may coincide if the graph is not biconnected). In the former case v is a *sink-switch*, in the latter a *source-switch*. Observe that a source (sink) of D_ϕ is source-switch (sink-switch) of all its incident faces; a vertex of D_ϕ that is not a source or a sink is a switch of all its incident faces except two.

Consider the labeling of the angles of D_ϕ induced by its upward embedding. Let v be a switch of a face f of D_ϕ, and let e_1 and e_2 be two consecutive edges on the boundary of f that are incident on v. Clearly, (e_1, e_2) is an angle of f. We call v an $s_{\mathtt{S}}$*-switch* ($s_{\mathtt{L}}$*-switch*) of f if v is a source-switch of f and if (e_1, e_2) is labeled `S` (`L`). We call v a $t_{\mathtt{S}}$*-switch* ($t_{\mathtt{L}}$*-switch*) of f if v is a sink-switch of f and if (e_1, e_2) is labeled `S` (`L`). Note that each `S` or `L` labels of a face is associated with a switch.

A *complete saturator* of D_ϕ is a set of vertices and edges, not belonging to D_ϕ, with which we augment D_ϕ. More precisely, a complete saturator consists of two vertices s and t, edge (s,t), and a set of edges (u,v) (each edge a *saturating edge*), such that (see Figure 5 (a)):

- vertices u and v are switches of the same face, or $u = s$ and v is an $s_{\mathtt{S}}$-switch of the external face, or u is a $t_{\mathtt{L}}$-switch of the external face and $v = t$,
- if $u, v \neq s, t$, either u is an $s_{\mathtt{S}}$-switch and v is an $s_{\mathtt{L}}$-switch or u is a $t_{\mathtt{L}}$-switch and v is a $t_{\mathtt{S}}$-switch; in the former case we say that u *saturates* v and in the latter case we say that v *saturates* u,
- the graph D_ϕ augmented with the vertices and the edges of the complete saturator is an upward embedded graph with an *st*-orientation (st-digraph); the upward embedding of such a digraph restricted to the vertices and the edges of D_ϕ coincides with $\mathcal{E}_\phi$.

A simple linear time algorithm for computing a complete saturator of D_ϕ is given in [1]. This algorithm works in two main steps:

In the first step it recursively decomposes each face f of G_ϕ adding a suitable number of saturating edges that split f. After this step, there are no more $s_{\mathtt{L}}$-switches and $t_{\mathtt{L}}$-switches in the internal faces of the digraph. Also, the $s_{\mathtt{L}}$-switches and $t_{\mathtt{L}}$-switches of the external face f are not alternated in the border of f.

In the second step the algorithm further decomposes the external face f, adding the vertices s, t and connecting s to every $s_{\mathtt{L}}$-switch of f, and t to every $t_{\mathtt{L}}$-switch of f.

In the following we briefly recall the algorithm for decomposing a face f of D_ϕ. More details can be found in [1]. We denote by σ_f the sequence of labels of the angles of f encountered in clockwise order while moving on the boundary of f. Also, we denote by $s_{\mathtt{L}}$ an `L` label of σ_f with associated a source-switch of f and by $t_{\mathtt{L}}$ an `L` label of σ_f with associated a sink-switch of f. Similarly, we use symbols $s_{\mathtt{S}}$ and $t_{\mathtt{S}}$ to denote `S`-labels with associated a source-switch of f and a sink-switch of f, respectively.

Algorithm *Saturate-Face(f)*

1. If f has exactly one source-switch and one sink-switch then return.
2. Find a subsequence (x, y, z) in σ_f such that x is an `L` label, and y and z are `S` labels. Let v_x, v_y, and v_z be the switches of f associated with x, y, and z, respectively.
3. Split f into two faces f' and f'' by inserting one edge; after the split, f'' always consists of the part of f containing v_x, v_y, and v_z plus the new edge; f'' has only one source and only one sink. Two cases are possible for the new edge:

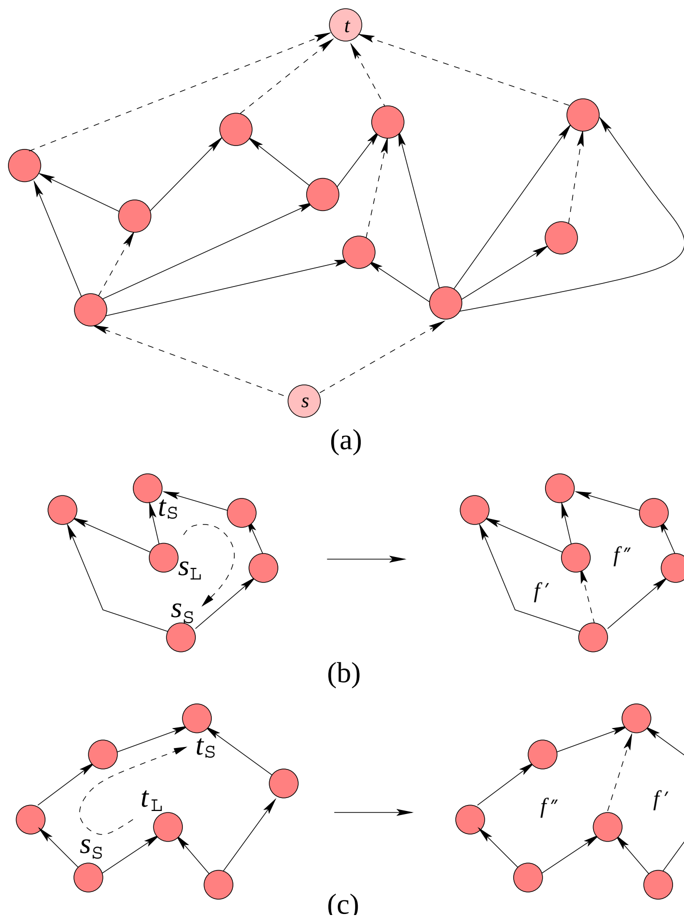


Figure 5: (a) An upward embedded digraph with a complete saturator. The edges of the saturator are dashed. (b) Illustration of Case 1 of algorithm *Saturate-Face(f)*. (c) Illustration of Case 2 of algorithm *Saturate-Face(f)*

Case 1 $(x, y, z) = (s_L, t_S, s_S)$: Add edge (v_z, v_x); f' consists of the part of f that does not contain v_y plus the new edge. Also, $\sigma_{f'}$ is obtained from σ_f by replacing the subsequence (x, y, z) with an s_S. (see Figure 5 (b)).

Case 2 $(x, y, z) = (t_L, s_S, t_S)$: Add edge (v_x, v_z); f' consists of the part of f that does not contain v_y plus the new edge. Also, $\sigma_{f'}$ is obtained from σ_f by replacing the subsequence (x, y, z) with a t_S. (see Figure 5 (c)).

4. Apply *Saturate-Face(f')*.

3 Characterizing Upward Embeddings

In this section we provide a complete characterization of the set of all upward embeddings of a general embedded planar graph (Section 3.1); it also implies a characterization of the upward orientations of the given graph. We model such a set of upward embeddings by using a simple network flow technique, which extends and generalizes that described by Bousset [3] for characterizing bipolar orientations. Also, we show how it is possible to add costs to our flow model in order to compute in polynomial time an upward orientation with the minimum number of sources and sinks (Section 3.2).

3.1 A Flow Model Characterizing Upward Embeddings

The following theorem characterizes the class of labelings that can be determined by any upward embedding of an embedded planar graph. It is important to observe that the characterization of such a class of labelings does not depend either on the choice of a splitting of the adjacency lists of the graph, in contrast to the result given in Lemma 1, or on the choice of an orientation of the graph.

Theorem 1 *Let $\mathcal{L}$ be any labeling of the angles of an embedded planar graph G_ϕ with labels* `L`*,* `S`*, and* `F`*. $\mathcal{L}$ is the labeling determined by an upward embedding of G_ϕ if and only if the following properties hold:*

(FIN') $deg(f) - 2 = 2L(f) + F(f)$*, for each internal face f of G_ϕ.*

(FEX') $deg(f) + 2 = 2L(f) + F(f)$*, for the external face f of G_ϕ.*

(VL) *For each vertex v either $F(v) = 2$ and $L(v) = 0$ or $F(v) = 0$ and $L(v) = 1$.*

Proof:
The necessary condition is an immediate consequence of Lemma 1 and Lemma 2. In fact, if $\mathcal{L}$ is determined by an upward embedding, then properties (FIN), (FEX), (VL0), and (VL1) of Lemma 1 hold. From Lemma 2 properties (FIN) and (FEX) are equivalent to properties (FIN') and (FEX'); further, properties (VL0) and (VL1) imply that one of the two cases of property (VL) holds, for each vertex of G_ϕ.

To prove the sufficiency of the condition we consider a labeling $\mathcal{L}$ that verifies properties (FIN'), (FEX'), and (VL), and construct an upward embedding of G_ϕ that determines $\mathcal{L}$. From $\mathcal{L}$, we construct a splitting $\mathcal{E}_\phi$ of the adjacency lists of G_ϕ as follows:

- We observe that there exists at least two distinct vertices s and t on the external face f having an angle labeled with `L`. In fact, from property (FEX') (that is equivalent to property (FEX) of Lemma 1) we must have that $L(f) = S(f) + 2$. We assign all the edges incident on s to the list $E_{above}(s)$ (we set $E_{below}(s)$ empty). Namely, if (e_1, e_2) is the angle with label `L` at vertex s, e_2 and e_1 will be the first edge and the last edge of $E_{above}(s)$, respectively.

- We execute a breadth first search starting from s. At each step we visit a different vertex v and split the list of the edges that are incident on v. In a breadth first search all the edges (and hence all the angles) incident on a vertex v are explored when v is visited. We chose to scan these edges in clockwise order. Namely, suppose that v is visited by moving from vertex u through edge $e_0 = (u, v)$ (e_0 is the parent edge of v in the breadth first search). If e_0 is in $E_{above}(u)$ we put e_0 in $E_{below}(v)$, while if e_0 is in $E_{below}(u)$ we put e_0 in $E_{above}(v)$. Suppose that $e_0, e_1, \ldots, e_k$ are the edges incident on v in this clockwise ordering. For each e_i ($i = 0, \ldots, k-1$) we consider the label l of angle (e_i, e_{i+1}), and we decide if e_{i+1} has to be assigned to $E_{above}(v)$ or to $E_{below}(v)$. Note that, at this point, e_i has been already assigned to one of the two lists. The following cases are possible: (1) If $l = \mathtt{L}$ and $e_i \in E_{below}(v)$ then e_{i+1} is put at the end of $E_{below}(v)$. (2) If $l = \mathtt{L}$ and $e_i \in E_{above}(v)$ then e_{i+1} is put at the start of $E_{above}(v)$. (3) If $l = \mathtt{S}$ and $e_i \in E_{below}(v)$ then e_{i+1} is put immediately before e_i in $E_{below}(v)$. (4) If $l = \mathtt{S}$ and $e_i \in E_{above}(v)$ then e_{i+1} is put immediately after e_i in $E_{above}(v)$. (5) If $l = \mathtt{F}$ and $e_i \in E_{below}(v)$ then e_{i+1} is put at the start of $E_{above}(v)$. (6) If $l = \mathtt{F}$ and $e_i \in E_{above}(v)$ then e_{i+1} is put at the end of $E_{below}(v)$.

It is easy to see that $\mathcal{E}_\phi$ verifies (E1). To prove that $\mathcal{E}_\phi$ is an upward embedding of G_ϕ we need only to prove that properties (VL0) and (VL1) of Lemma 1 are verified (since properties (FIN) and (FEX) are equivalent to properties (FIN') and (FEX')). From property (VL) we only have two possible cases for the labels of the angles at each vertex v of G_ϕ.

- $F(v) = 2$ and $L(v) = 0$. This implies that, for splitting the edges incident on v cases (1) and (2) are never applied, cases (5) and (6) are applied twice in the total, and cases (3) and (4) are applied $deg(v) - 2$ times in the total. Also, cases (5) and (6) imply that neither $E_{above}(v)$ nor $E_{below}(v)$ will be empty. This matches property (VL0).

- $F(v) = 0$ and $L(v) = 1$. This implies that, for splitting the edges incident on v, either case (1) or case (2) is applied once, cases (5) and (6) are never applied, and either case (3) or case (4) is applied $deg(v) - 1$ times. Also, observe that each of the cases (1), (2), (3), and (4) always puts e_{i+1} in the same list as e_i, and that either (1) and (3) or (2) and (4) are applied. This guarantees that exactly one of the two lists $E_{above}(v)$ and $E_{below}(v)$ will be empty. This matches property (VL1).

Finally, since no other cases are possible, properties (VL0) and (VL1) of Lemma 2 hold.

q.e.d.

We call *upward labeling* of G_ϕ a labeling of the angles of G_ϕ that verifies properties (FIN'), (FEX'), and (VL) of Theorem 1. The result of Theorem 1 allows the description of all upward embeddings of G_ϕ in terms of upward labelings of G_ϕ. Note that, the proof of the theorem provides a procedure to construct the upward embedding associated with an upward labeling. Actually, for each upward labeling, there are exactly two "symmetric" upward embeddings that determine it; they are obtained one from the other by simply exchanging list $E_{above}(v)$ with list $E_{below}(v)$ for each vertex v and then reversing such lists (see Figure 7 (b) for an example).

We now provide a network flow model that characterizes all the upward labelings of G_ϕ. Because of the above considerations, this flow model provides a characterization of all upward embeddings of G_ϕ. We associate with G_ϕ a flow network $\mathcal{N}_\phi$, such that the integer feasible flows on $\mathcal{N}_\phi$ are in one-to-one correspondence with the upward labelings of G_ϕ. Flow network $\mathcal{N}_\phi$ is a directed graph defined as follows (see Figure 6):

- The nodes of $\mathcal{N}_\phi$ are the vertices (*vertex-nodes*) and the faces (*face-nodes*) of G_ϕ. Each vertex-node supplies flow 2 and each face-node associated with face f of G_ϕ demands a flow equal to $deg(f) - 2$ if f is internal and $deg(f) + 2$ if f is external.
- With each angle of G_ϕ at vertex v in face f there is an associated arc (v, f) of $\mathcal{N}_\phi$ with lower capacity 0 and upper capacity 2.

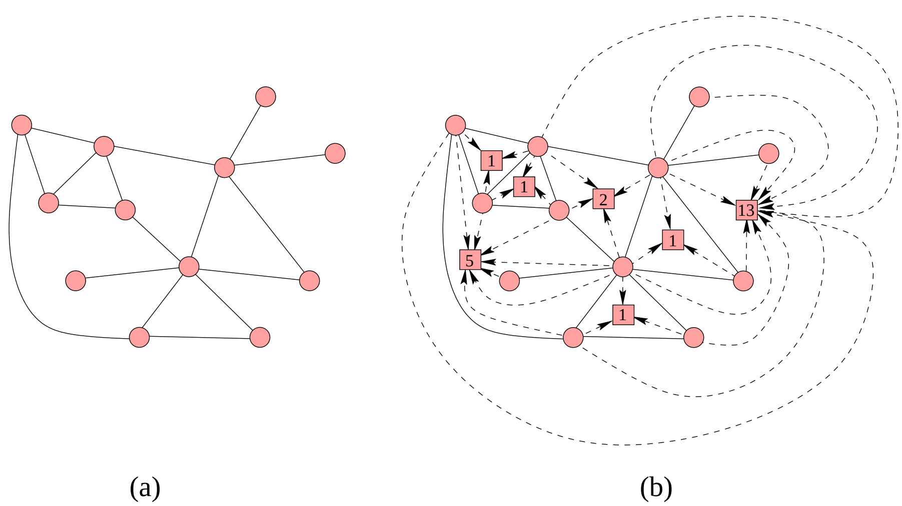


Figure 6: (a) An embedded planar graph G_ϕ. (b) Flow network $\mathcal{N}_\phi$ associated with G_ϕ. The vertex-nodes are circles and the face-nodes are squares. Each face-node is labeled with its demand. The arcs of the networks are dashed.

Observe that in $\mathcal{N}_\phi$ the total demand is equal to the total supply. In fact:

$$\sum_{f \in F} (deg(f) - 2) + 4 = \sum_{f \in F} deg(f) - 2|F| + 4 = 2|E| - 2|F| + 4 = 2|V|.$$

The intuitive interpretation of the flow model in terms of upward embedding is as follows: (i) Each unit of flow represents a flat angle, with the convention that a large angle counts as two flat angles; an arc a of $\mathcal{N}_\phi$ has flow $0, 1$, or 2, depending on the fact that its associated angle is small, flat, or large, respectively. (ii) The demand of each face-node and the supply of each vertex-node reflect the balancing properties (FIN'), (FEX') and (VL). Figure 7 shows a feasible flow on the network associated with an embedded planar graph, the corresponding upward labeling, and the two "symmetric" upward embeddings associated with the labeling. Theorem 2 formally proves the correctness of the intuitive interpretation described above .

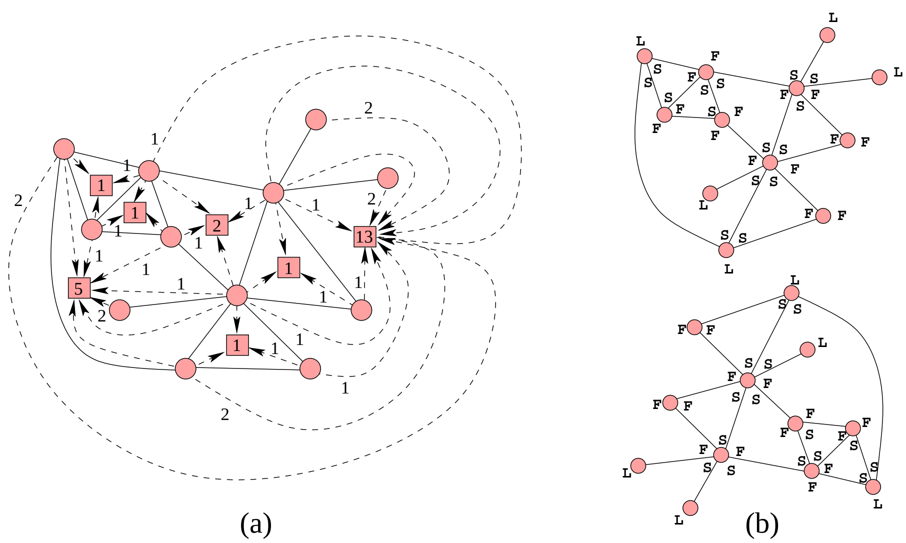


Figure 7: (a) A feasible flow on the network associated with an embedded planar graph. Only the flow values different from zero are shown. (b) The upward labeling $\mathcal{L}$ corresponding to the flow and the two "symmetric" upward embeddings associated with $\mathcal{L}$.

We remark that network $\mathcal{N}_\phi$ is related to the flow model used by Bousset for describing bipolar orientations of biconnected embedded planar graphs. The flow values in such a model do not allow to represent large angles (the allowed flow values are only 0 or 1), and the source and the sink of the final orientation are prescribed. Our flow model extends and generalizes the model of Bousset to 1-connected planar graphs, by allowing the representation of any kind of upward orientations and embeddings, including the bipolar orientations for biconnected graphs.

Theorem 2 *Let G_ϕ be an embedded planar graph and let $\mathcal{N}_\phi$ be the flow network associated with G_ϕ. There is a one-to-one correspondence between the set of the upward labelings of G_ϕ and the set of the integer feasible flows on $\mathcal{N}_\phi$.*

Proof:
Consider an upward labeling $\mathcal{L}$ of G_ϕ. From it we construct an integer feasible flow x of $\mathcal{N}_\phi$ as follows. For each angle α of G_ϕ let a be the arc of $\mathcal{N}_\phi$ associated with α. We set $x(a) = 2$ if α is labeled `L`, $x(a) = 1$ if α is labeled `F`, and $x(a) = 0$ if α is labeled `S`. The above construction is clearly an injective transformation. In fact, there is a one-to-one correspondence between angles of G_ϕ and arcs of $\mathcal{N}_\phi$ and hence, different labelings of the same angle of G_ϕ produces different values of flow on the corresponding arc of $\mathcal{N}_\phi$. We now prove that flow x is feasible. From the construction of x and from property (VL) of $\mathcal{L}$, it follows that every vertex-node of $\mathcal{N}_\phi$ supplies flow 2 (and demands flow 0). Hence, the balance property of x on every vertex-node of $\mathcal{N}_\phi$ is verified. Let f be an internal face of G_ϕ, and consider the face-node of $\mathcal{N}_\phi$ associated with f. From the construction of x, such a face-node receives a flow equal to $2L(f)+F(f)$ and supplies flow 0 ; hence, from property (FIN') of $\mathcal{L}$, it demands a flow equal to $deg(f) - 2$. The same reasoning applies for the external face, using property (FEX'). Hence, also the balance property of x on every face-node is verified. Finally, since on each arc of $\mathcal{N}_\phi$ we assign an integer amount of flow in the range $[0, 2]$, the lower and upper capacities on the arcs of $\mathcal{N}_\phi$ are respected by x.

Conversely, consider an integer feasible flow x of $\mathcal{N}_\phi$, and construct from x a labeling $\mathcal{L}$ of G_ϕ, by applying a transformation that is the reverse of that described above . Namely, for each arc a of $\mathcal{N}_\phi$ denote by α the corresponding angle of G_ϕ. Labeling $\mathcal{L}$ is constructed by assigning label `L`, `F`, and `S` to α, depending on the case that $x(a) = 2$, $x(a) = 1$, and $x(a) = 0$, respectively. By using the properties of x and the same reasoning applied above, it is easy to prove that $\mathcal{L}$ is an upward labeling of G_ϕ. *q.e.d.*

Theorem 1 and Theorem 2 allow us to compute an upward embedding of an embedded planar graph G_ϕ by computing an integer feasible flow on network $\mathcal{N}_\phi$. We now analyze the running time complexity of computing an upward embedding by means of a flow technique.

Network $\mathcal{N}_\phi$ has $O(n)$ vertices and edges, where n denotes the number of vertices of G_ϕ. Both $\mathcal{N}_\phi$ and an upward embedding associated with a feasible flow on $\mathcal{N}_\phi$ can be constructed in linear time. We now observe that $\mathcal{N}_\phi$ can be easily reduced to an equivalent unit capacity network $\mathcal{N}_\phi^*$ with a single source s and a single sink t and with $O(n)$ nodes and arcs. On $\mathcal{N}_\phi^*$ we can apply Dinic's algorithm to compute in $O(n^{3/2})$ time a feasible (maximum) flow [7]. Namely, $\mathcal{N}_\phi^*$ is obtained from $\mathcal{N}_\phi$ by replacing each arc a with two unit capacity arcs having the same direction as a, by connecting s to each vertex-node with two unit capacity arcs, by connecting each internal face-node f to t with $deg(f) - 2$ unit capacity arcs, and by connecting the external face-node h to t with $deg(h)+2$ unit capacity arcs. Finally, node s supplies flow $2|V|$ and node t demands flow $2|V|$, while all the other nodes demand and supply flow 0. The following theorem summarizes the complexity analysis.

Theorem 3 *There exists a flow technique for computing an upward embedding* *of an undirected embedded planar graph in $O(n^{3/2})$ time and $O(n)$ space, where n denotes the number of vertices of the graph.*

There are two main advantages of computing upward embeddings of a general planar graph G_ϕ by using the flow model described so far. First, no augmentation algorithm has to be used to make the input graph biconnected (we just apply a standard flow algorithm). Second, it is possible to deal with partially specified embeddings. In particular it is possible to constrain an angle to be large by fixing flow 2 on the corresponding arc of the network and to constrain a vertex to be neither a source nor a sink by reducing to 1 the upper capacity of its leaving arcs in the network. Also observe that in the presence of constraints a feasible solution might not exist, and in this case a feasible flow is not found.

In the next section we describe how to compute upward embeddings with the minimum number of sources and sinks, by adding costs to our network.

3.2 Minimizing Sources and Sinks

Computing an upward embedding of G_ϕ with the minimum number of sources and sinks (which we call *optimal upward embedding* for simplicity) is equivalent to computing an upward embedding with the minimum number of large angles. Clearly, if the graph is biconnected, the problem is reduced to the computation of a bipolar orientation. For this reason, we regard the concept of optimal upward embedding as the natural extension of the definition of bipolar orientation to the case of general connected graphs.

The flow model we use to compute an optimal upward orientation of G_ϕ is a simple variation of the one described for characterizing upward embeddings (see Section 3.1). We add a linear number of arcs to network $\mathcal{N}_\phi$ and we equip the arcs of the new network with costs. Each unit of cost represents a large angle. We also reduce the upper capacity of all the arcs of the network. More in detail, the new network $\widetilde{\mathcal{N}_\phi}$ is derived from $\mathcal{N}_\phi$ as follows: for each angle of G_ϕ at vertex v in face f we substitute its associated arc in $\mathcal{N}_\phi$ with a pair of directed arcs $a_v = (v, f), a'_v = (v, f)$. Both the new arcs have lower capacity 0 and upper capacity 1. Also, arc a_v has cost 0 while arc a'_v has cost 1.

Let x be a minimum cost flow on $\widetilde{\mathcal{N}_\phi}$. The interpretation of the flow in terms of upward labeling is similar to the one given for $\mathcal{N}_\phi$, with a slight variation due to the additional arcs and costs. We first observe that for each pair of arcs a_v, a'_v it never happens $x(a_v) = 0$ and $x(a'_v) = 1$, due to the fact that the cost of a_v is 0 and that the cost of a'_v is 1. In fact, if $x(a_v) = 0$ and $x(a'_v) = 1$, then there would exist a negative cost cycle represented by the two arcs a'_v, a_v, and it would be possible to derive a new flow x' from x by simply exchanging one unit of flow between a'_v and a_v (i.e., $x'(a_v) = 1$ and $x'(a'_v) = 0$). This would imply that x' has a cost smaller than the cost of x, in contrast to the assumption that x has the minimum cost. Hence, the only possibilities for the flow on arcs a_v, a'_v are: (i) $x(a_v) = x(a'_v) = 0$, the angle associated with arcs a_v, a'_v is small.

(ii) $x(a_v) = 1$ and $x(a'_v) = 0$, the angle associated with arcs a_v, a'_v is flat. (iii) $x(a_v) = x(a'_v) = 1$, the angle associated with arcs a_v, a'_v is large.

Note that, only in the third case we have cost 1 on arcs a_v, a'_v, while in the other two cases we have cost 0. This implies that the total cost of flow x on $\widetilde{\mathcal{N}_\phi}$ represents the total number of large angles of the corresponding upward embedding of G_ϕ. Hence, since x has the minimum cost, the corresponding upward embedding has the minimum number of large angles.

Let n be the number of vertices of G_ϕ. Since network $\widetilde{\mathcal{N}_\phi}$ is planar and has $O(n)$ vertices, and since its total demand (supply) is $O(n)$, a minimum cost flow on $\widetilde{\mathcal{N}_\phi}$ can be computed in $O(n^{\frac{7}{4}} \log n)$ time by the algorithm described in [10]. The following theorem summarizes the main contribution of this section.

Theorem 4 *There exists an $O(n^{\frac{7}{4}} \log n)$ time algorithm that computes an upward embedding of an embedded 1-connected planar graph with the minimum number of sources and sinks.*

We conclude this section by giving an upper bound on the number of sources and sinks of an optimal upward embedding.

Lemma 3 *An optimal upward embedding of an embedded planar graph G_ϕ has at most $B + 1$ sources and sinks, where B is the number of blocks of G_ϕ.*

Proof: We prove the lemma by induction on B. If $B = 1$, the graph is biconnected and an optimal upward embedding of it has exactly one source and one sink. Suppose that the lemma is true for each graph with $B \geq 1$ blocks, and consider a graph G_ϕ with $B + 1$ blocks. We select any block C of G_ϕ such that C contains exactly one cutvertex of G_ϕ and there is no block nested into C. Note that such a block always exists. Let $G'_{\phi'}$ be the graph obtained from G_ϕ by removing C and let $\mathcal{E}'_{\phi'}$ be an optimal upward embedding of $G'_{\phi'}$. From the inductive hypothesis, $\mathcal{E}'_{\phi'}$ has at most $B+1$ sources and sinks. From $\mathcal{E}'_{\phi'}$ we construct an upward embedding of G_ϕ. Such an upward embedding coincides with $\mathcal{E}'_{\phi'}$ for the subgraph $G'_{\phi'}$ and it is determined on C as follows. We always embed C above or below its cutvertex v, according to $\mathcal{E}'_{\phi'}$ and according to the planar embedding of G_ϕ. Namely, let e_1 and e_2 be the two edges (not necessarily distinct) of G_ϕ encountered immediately before and after C in the clockwise ordering around v. Three distinct cases are possible for $\mathcal{E}'_{\phi'}$:

- If both e_1 and e_2 belong to $E_{above}(v)$, we compute an upward embedding of C with exactly one source and one sink, where the source is v, and we embed it above v in $\mathcal{E}'_{\phi'}$ (see Figure 8 (a)).
- If both e_1 and e_2 belong to $E_{below}(v)$, we compute an upward embedding of C with exactly one source and one sink, where the sink is v, and we embed it below v in $\mathcal{E}'_{\phi'}$ (see Figure 8 (b)).
- If one between e_1 and e_2 belongs to $E_{above}(v)$ while the other edge belongs to $E_{below}(v)$, we arbitrarily choose to compute an upward embedding of C with exactly one source and one sink, where the source is v, and we embed it above v in $\mathcal{E}'_{\phi'}$ (see Figure 8 (c)).

The obtained upward embedding has at most one source or one sink more than $\mathcal{E}'_{\phi'}$, since vertex v is in common between C and $G'_{\phi'}$. Therefore, an optimal upward embedding of G_ϕ has at most $B+2$ sources and sinks.

q.e.d.

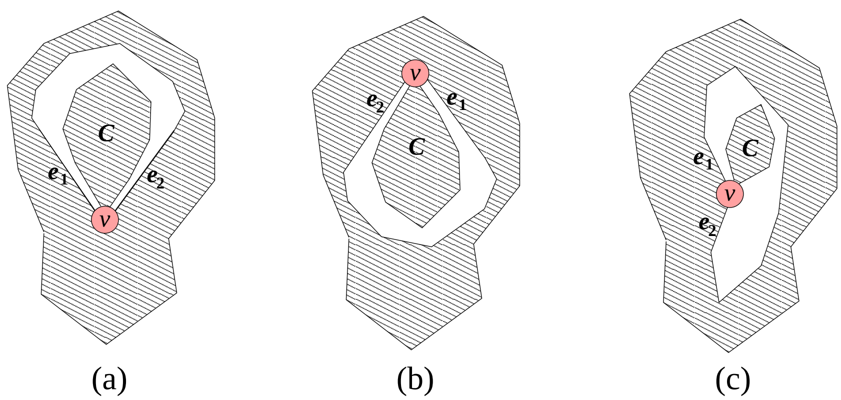


Figure 8: Illustration of the proof of Lemma 3.

The bound of Lemma 3 is strict and a class of plane graphs whose upward embeddings have $B+1$ sources and sinks can be obtained by nesting each block into another, as shown by the example of Figure 9.

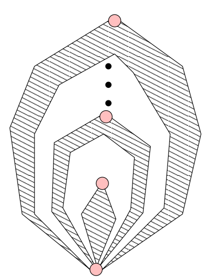


Figure 9: A class of embedded planar graphs whose optimal upward embeddings have $B+1$ sources and sinks (circles).

4 Algorithms for Visibility Representations

We use the above results on upward embeddings to compute drawings of general connected planar graphs. Namely, we focus on graph drawing algorithms which require the computation of a (*weak-*)*visibility representation* of the input graph as a preliminary step [6]. In a visibility representation (see Figure 10), each vertex is mapped to a horizontal segment and each edge (u, v) is mapped to a vertical segment between the segments associated with u and v; horizontal segments do not overlap, and each vertical segment only intersects its extreme horizontal segments.

A standard technique [6] for constructing a visibility representation of a planar graph G first computes a bipolar orientation of G and then computes the coordinates of the drawing from this orientation. If G is not biconnected the technique needs to augment the graph to a biconnected planar one, in order to compute a bipolar orientation of it. The augmentation algorithm adds to G a suitable number of dummy edges, which will be removed in the final drawing. However, this technique has several drawbacks: (i) Adding too many dummy edges may lead to a final drawing with area much bigger than necessary. On the other side, the problem of adding the minimum number of edges to make a planar graph biconnected and still planar is NP-hard [12]. (ii) Although a good approximation algorithm for the above augmentation problem exists [8] (which reaches the optimal solution in many cases), implementing it efficiently is quite difficult, because it requires us to deal with the *block cutvertex tree* [11] of the graph and with an efficient incremental planarity testing algorithm. In fact, such an approximation algorithm has $O(n^2 T)$ running time, where T is the amortized time bound per query or insertion operation of the incremental planarity testing algorithm. (iii) The presence of dummy edges in the graph makes difficult to handle with partial assignments of the upward embedding.

Tamassia and Tollis [17] provide a different linear time algorithm for computing visibility representations of general connected graphs. At each step of the algorithm a visibility representation of a new distinct block of the graph is computed and suitably merged to the current drawing. However, merging operations require the execution of scaling down geometric operations, which may lead to a final drawing with a big area on an integer grid. Also, the algorithm has many degrees of freedom about how to perform some topological operations and about the choice of the ordering in which the blocks are considered; different decisions may lead to very different results.

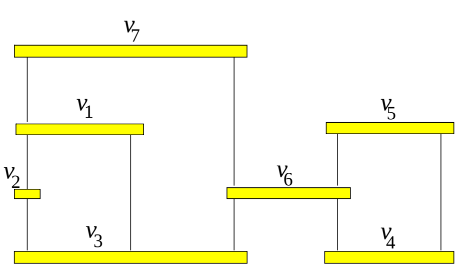


Figure 10: A visibility representation of the upward embedded graph shown in Figure 1(b).

We propose the following algorithm for computing a visibility representation of a 1-connected embedded planar graph G_ϕ.

Algorithm *Visibility-Upward-Embedding*

1. Compute an upward embedding $\mathcal{E}_\phi$ of G_ϕ by calculating a feasible flow on network $\mathcal{N}_\phi$.

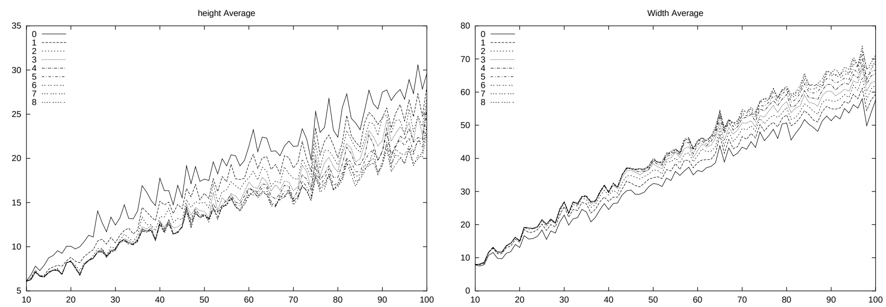
2. Compute an upward embedded *st*-digraph S_ϕ including G_ϕ and preserving $\mathcal{E}_\phi$ on G_ϕ, by using the linear time saturation procedure described at the end of Section 2.

3. Compute a visibility representation of S_ϕ (within its upward embedding) by using any known linear time algorithm [6], and then remove the edges introduced by the saturation procedure.

Algorithm *Visibility-Upward-Embedding* has $O(n^{3/2})$ running time, because its time complexity is dominated by the cost of computing a feasible flow on $\mathcal{N}_\phi$. We experimentally observed that the area of the visibility representations produced by this algorithm can be dramatically improved by computing upward embeddings with the minimum number of sources and sinks. To do that we just apply a min-cost-flow algorithm in Step 1. Clearly, in this case, the running time of the whole algorithm grows to $O(n^{\frac{7}{4}} \log n)$.

We have also slightly refined Algorithm *Visibility Upward Embedding* aiming to get a certain control over the width and the height of visibility representations of 1-connected planar graphs. After we have computed an upward embedding with the minimum number of switches we rearrange the blocks around the cutvertices in the upward embedding. Namely, if v is a cutvertex we place all the blocks of v either above or below. This often leads to a reduction of the height and to an increase in the width. Such a rearrangement is performed in linear time by exploiting the flow network associated with the embedded planar graph. We experimented such an approach on a randomly generated test suite of 1820 graphs whose number n of vertices ranges from 10 to 100 (20 instances for each value of n). A detailed description of the procedure used to generate the graphs can be found in [15]. We averaged the width and the height on all the graphs having the same number of vertices. Charts in Figure 11 graphically show the results of the experimentation for the maximum number of cutvertices $k (k = 0 \ldots 8)$ whose blocks have been rearranged.

Also, Figure 12 compares the area of the drawings computed with this strategy, where k is chosen equal to the total number of cutvertices of the graph, against the area of the drawings computed with a standard technique which uses the approximation algorithm in [8] to initially make the graph biconnected. In the two strategies we use the same algorithm for constructing the visibility representation from the *st*-digraph. Experimentally, for the considered test suite, the running time of the two algorithms is comparable (less than one second for the largest graphs).



(a) height (b) width

Figure 11: The charts show how rearranging the blocks around cutvertices affects the width and the height of the visibility representation.

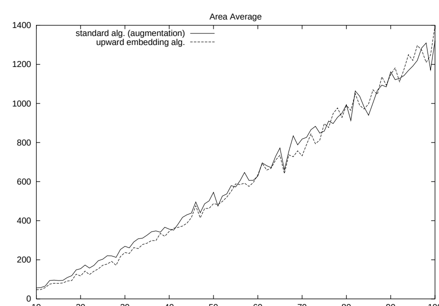


Figure 12: Area of the drawings computed with our strategy against the area of the drawings computed with a standard technique based on a sophisticated augmentation algorithm (average values). The x-axis represents the number of vertices.

5 Open Problems

There are several open problems that we plan to study in the near future. For example, we are interested in an algorithm for counting and enumerating all upward embeddings of an embedded planar graph without repetitions. Also, is it possible to pass from an upward embedding to any other in linear time? Is there a linear time algorithm to compute optimal upward embeddings of embedded planar graphs? What about non-embedded planar graphs? Finally, from an applications point of view we believe that the techniques shown in this paper may be successfully refined to compute drawings that approximate a given width/height ratio.

References

[1] P. Bertolazzi, G. Di Battista, G. Liotta, and C. Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 6(12):476–497, 1994.

[2] P. Bertolazzi, G. Di Battista, C. Mannino, and R. Tamassia. Optimal upward planarity testing of single-source digraphs. *SIAM J. Comput.*, 27(1):132–169, 1998.

[3] M. Bousset. A flow model of low complexity for twisting a layout. In *Workshop on Graph Drawing (GD'93)*, pages 43–44, 1993.

[4] J. Czyzowicz, A. Pelc, and I. Rival. Drawing orders with few slopes. Technical Report TR-87-12, Department of Computer Science, University of Ottawa, 1987.

[5] H. de Fraysseix, P. O. de Mendez, and P. Rosenstiehl. Bipolar orientations revisited. *Discrete Appl. Math.*, 56:157–179, 1995.

[6] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing*. Prentice Hall, Upper Saddle River, NJ, 1999.

[7] S. Even and R. E. Tarjan. Network flow and testing graph connectivity. *SIAM J. Comput.*, 4:507–518, 1975.

[8] S. Fialko and P. Mutzel. A new approximation algorithm for the planar augmentation problem. In *Symposium on Discrete Algorithms (SODA'98)*, pages 260–269, 1998.

[9] A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing (Proc. GD '94)*, volume 894 of *Lecture Notes Comput. Sci.*, pages 286–297. Springer-Verlag, 1995.

[10] A. Garg and R. Tamassia. A new minimum cost flow algorithm with applications to graph drawing. In S. C. North, editor, *Graph Drawing (Proc. GD '96)*, volume 1190 of *Lecture Notes Comput. Sci.*, pages 201–216. Springer-Verlag, 1997.

[11] F. Harary. *Graph Theory*. Addison-Wesley, Reading, MA, 1972.

[12] G. Kant and H. L. Bodlaender. Planar graph augmentation problems. In *Proc. 2nd Workshop Algorithms Data Struct.*, volume 519 of *Lecture Notes Comput. Sci.*, pages 286–298. Springer-Verlag, 1991.

[13] D. Kelly. Fundamentals of planar ordered sets. *Discrete Math.*, 63:197–216, 1987.

[14] D. Kelly and I. Rival. Planar lattices. *Canad. J. Math.*, 27(3):636–665, 1975.

[15] M. Pizzonia. Engineering of graph drawing algorithms for applications. *PhD thesis*, 2001. Dipartimento di Informatica e Sistemistica, Università "La Sapienza" di Roma.

[16] I. Rival. Reading, drawing, and order. In I. G. Rosenberg and G. Sabidussi, editors, *Algebras and Orders*, pages 359–404. Kluwer Academic Publishers, 1993.

[17] R. Tamassia and I. G. Tollis. A unified approach to visibility representations of planar graphs. *Discrete Comput. Geom.*, 1(4):321–341, 1986.

[18] R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.