Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 9, no. 3, pp. 365–389 (2005)

Computing Radial Drawings on the Minimum Number of Circles

Emilio Di Giacomo *Walter Didimo* *Giuseppe Liotta*

Dipartimento di Ingegneria Elettronica e dell'Informazione,
Università degli Studi di Perugia, Perugia, Italy
http://www.diei.unipg.it/∼digiacomo http://www.diei.unipg.it/∼didimo
http://www.diei.unipg.it/∼liotta
digiacomo@diei.unipg.it didimo@diei.unipg.it liotta@diei.unipg.it

Henk Meijer

School of Computing, Queen's University, Kingston, Ontario, Canada
http://www.cs.queensu.ca/∼henk/
henk@cs.queensu.ca

Abstract

A radial drawing is a representation of a graph in which the vertices lie on concentric circles of finite radius. In this paper we study the problem of computing radial drawings of planar graphs by using the minimum number of concentric circles. We assume that the edges are drawn as straight-line segments and that co-circular vertices can be adjacent. It is proven that the problem can be solved in polynomial time. The solution is based on a characterization of those graphs that admit a crossing-free straight-line radial drawing on k circles. For the graphs in this family, a linear time algorithm that computes a radial drawing on k circles is also presented.

Article Type	Communicated by	Submitted	Revised
Regular Paper	E. R. Gansner and J. Pach	January 2005	July 2005

Research partially supported by MIUR under Project ALGO-NEXT (Algorithms for the Next Generation Internet and Web: Methodologies, Design and Experiments), and by NSERC. An extended abstract of this paper was presented at 12th *International Symposium on Graph Drawing*, GD 2004.

1 Introduction

A radial drawing is a representation of a graph in which the vertices are constrained to lie on concentric circles of finite radius. Drawing graphs radially is relevant in situations where it is important to display a graph with the constraint that some vertices are drawn "more central" than others. Examples of such applications include social networks analysis (visualization of policy networks and co-citation graphs), operating systems (visualization of filesystems), cybergeography (visualization of Web maps and communities), and bioinformatics (visualization of protein-protein interaction diagrams); see e.g. [5, 11, 12].

This paper investigates crossing-free radial drawings of planar graphs. Let G be a planar graph. A crossing-free radial drawing of G induces a partition of its vertices into levels such that vertices in the same level are co-circular in the drawing; for each level, the planarity of the drawing induces a circular ordering of the vertices in the level. Conversely, in order to construct a radial drawing of G a partition of its vertices into levels and a circular ordering within each level must be found such that vertices of the same level are drawn co-circularly and the edges can be drawn without intersecting each other.

Bachmaier, Brandenburg, and Forster [1, 2] investigate the radial planarity testing problem: Given a partition of the vertices of G into levels, they want to test whether there exists a crossing-free radial drawing of G consistent with the given leveling (i.e. vertices in the same level can be drawn on the same circle and the edges can be added without crossing). In [1] it is assumed that the edges are drawn as strictly monotone curves from inner to outer circles and that no two co-circular vertices are connected by an edge. The elegant linear-time algorithm presented by Bachmaier, Brandenburg, and Forster tests radial planarity by using an extension of PQ-trees, called PQR-trees. In [2] the authors extend the algorithm to the case where edges between consecutive co-circular vertices are allowed.

In this paper we assume that the partition of the vertices of G is not given and study the problem of computing a partition that minimizes the number of levels, i.e. that corresponds to a crossing-free straight-line radial drawing of G on the minimum number of circles. We call such a drawing a *minimum radial drawing* of G. We assume that the edges are straight-line segments and that vertices on the same level can be adjacent. These choices are justified by different application-oriented examples of radial drawings that adopt the straight-line standard (see e.g. [6, 17, 19]) and by the observation that allowing edges among co-circular vertices appears to be a natural approach for the reduction of the number of levels. We also remark that the study of crossing-free straight-line layered drawings in which vertices lie on parallel lines and where intra-layer edges are allowed has been receiving increasing interest in the recent literature (a limited list of citations includes [3, 7, 13, 18]).

The contribution of the present paper is to characterize those graphs that can be drawn on a given number of concentric circles and to use this characterization to solve the optimization problem described above. More precisely:

- We show that every 2-outerplanar graph admits a crossing-free straight-line radial drawing on two circles. The proof is constructive and the radial drawing can be computed in linear time. Preliminary results on computing radial drawings of 2-outerplanar graphs appear in [9].
- We generalize these results and characterize the family of graphs that admit a crossing-free straight-line radial drawing on at most $k \geq 2$ circles. We recall that similar characterization problems for straight-line k-layered drawings are studied for the case of $k \leq 3$; see, e.g. [7]. We also recall that every planar graph admits a crossing-free drawing with all vertices on the same circle and at most one bend per edge [10].
- Based on the characterization above, we show that there exists a polynomial time algorithm to compute a minimum radial drawing of a planar graph. The drawing has the additional property of being "proper", i.e. an edge always connects either co-circular vertices or vertices on consecutive circles. This contrasts with a result by Heath and Rosenberg [16] who prove that it is NP-complete to decide whether a planar graph admits a proper crossing-free layered drawing with vertices on parallel straight lines and no intra-layer edges.

The remainder of this paper is organized as follows. Preliminaries and an overview of our approach are given in Sections 2 and 3. In Section 4 we introduce the concept of canonical k-outerplanar graphs. Section 5 defines the equipped block cut-vertex tree data structure. An algorithm for drawing 2-outerplanar graphs on two circles is described in Section 6. In Section 7 the results of Section 6 are used to compute radial drawings of k-outerplanar graphs and to devise a polynomial time algorithm for proper radial drawings of planar graphs on the minimum number of circles. Conclusions and open problems can be found in Section 8.

2 Preliminaries

We use the basic definitions regarding graph drawing from [8]. Let G be a graph. A *drawing* Γ of G maps each vertex v of G to a distinct point $p(v)$ of the plane and each edge $e = (u, v)$ of G to a simple Jordan curve connecting $p(u)$ and $p(v)$ such that each edge intersects no vertex except its endpoints. Drawing Γ is *planar* if no two distinct edges intersect except at common endvertices. Graph G is *planar* if it admits a planar drawing. A planar drawing Γ of G partitions the plane into topologically connected regions called the *faces* defined by Γ. The unbounded face is called the *external face*. The *boundary* of a face is its delimiting circuit (not necessarily a simple cycle) described by the circular list of its edges and vertices. The *boundary* of the external face, also called the *external boundary*, is the circular list of edges and vertices delimiting the unbounded region. If the graph is biconnected the boundary of each face is a simple cycle. An *embedding* of a planar graph G is an equivalence class of planar drawings that define the same set of faces, that is, the same set of face boundaries. A planar graph G with a given embedding Ψ is called an *embedded planar graph*. A drawing Γ of G is an *embedding preserving drawing* if $\Gamma \in \Psi$.

A 1-*outerplanar embedded graph* (also called 1-*outerplane graph*) is an embedded planar graph where all vertices are on the external face. An embedded graph is a k-*outerplanar embedded graph* (also called k-*outerplane graph*) ($k > 1$) if the embedded graph obtained by removing all vertices of the external face is a $(k-1)$-outerplane graph. The planar embedding of a k-outerplane graph is called a k-*outerplanar embedding*. A graph is k-*outerplanar* if it admits a k-outerplanar embedding. A planar graph G has *outerplanarity* k (for an integer $k > 0$) if it is k-outerplanar and it is not j-outerplanar for $0 < j < k$.

Let G be a k-outerplane graph with $k > 1$. We associate a *level* with each vertex v of G, denoted as $lev(v)$, according to the following definition: $lev(v) = 0$ if v is on the external face of G and $lev(v) = i$ ($i = 1, \ldots, k-1$) if v is on the external face after the removal of every vertex u with $lev(u) < i$. If $lev(v) = i$, we say that v is a vertex of level i. Let V_i be the set of vertices v with $lev(v) = i$. The subgraph induced by V_i is denoted by $G_i = (V_i, E_i)$. Notice that G_i is a graph of outerplanarity 1. Let $V_{i,i+1} = V_i \cup V_{i+1}$. The subgraph induced by $V_{i,i+1}$ is denoted by $G_{i,i+1} = (V_{i,i+1}, E_{i,i+1})$. We denote as $\overline{E}_{i,i+1}$ the set of edges that have an end-vertex of level i and an end-vertex of level $i+1$, i.e. $\overline{E}_{i,i+1} = E_{i,i+1} \setminus (E_i \cup E_{i+1})$.

We use $C_0, C_1, \ldots, C_{k-1}$ to denote a set of k concentric circles in the plane, where the radius of C_i is greater than the radius of C_{i+1} ($i = 0, \ldots, k-2$). Let G be a planar graph and let Γ be a crossing-free straight-line drawing of G. The drawing Γ is a *radial drawing* if the vertices of G are placed on a set of concentric circles. Γ will be called a k-*radial drawing* of G if it is a radial drawing on $C_0, C_1, \ldots, C_{k-1}$. Γ is a *minimum radial drawing* if it uses the minimum number of circles. If all edges of a radial drawing Γ connect either vertices on the same circle or vertices on consecutive circles, Γ is called a *proper radial drawing*.

Let G be a k-outerplane graph. A radial drawing of G is *level-preserving* if it is a k-radial drawing and every vertex v with $lev(v) = i$ is drawn on circle C_i. Observe that a level-preserving k-radial drawing of a k-outerplane graph is proper.

3 Overview of the Approach

We study the problem of computing a radial drawing of a planar graph G on the minimum number of circles. Our approach is as follows:

- We prove that if a graph has outerplanarity k then it admits a k-radial drawing.

- We prove that if a graph has a radial drawing on k-circles then it has outerplanarity at most k.

- We use the above characterization and a result by Bienstock and Monma [4] to show that there exists an $O(n^5 \log n)$-time algorithm that computes a minimum radial drawing of G.

The trickiest part is to show that a graph with outerplanarity k has a k-radial drawing. To this end, we provide a linear-time algorithm that receives as input a k-outerplane graph G and computes a level-preserving k-radial drawing of G. We start with G_0 and draw the vertices of V_0 on C_0 while maintaining their circular ordering in G_0. After placing the vertices of V_i on C_i we compute the radius of C_{i+1} and draw the vertices of V_{i+1} on C_{i+1} without moving any vertex from V_j with $0 \leq j \leq i$. For ease of presentation, we will define *canonical* k-outerplanar graphs and show how each k-outerplane graph can be transformed into a canonical k-outerplane graph. We will also show that a k-outerplane graph has a k-radial drawing if and only if its canonical form has a k-radial drawing.

4 Canonical Graphs

Let G be a k-outerplane graph. A *mixed face* of G is a face containing vertices of two consecutive levels. G is called *inter-triangulated* if all its mixed faces are three-cycles. Assume G is inter-triangulated and let c be a cut-vertex of G_{i+1} ($0 \leq i \leq k-2$). Let B and B' be two blocks (i.e. biconnected components) of G_{i+1} that are consecutive when going around c in clockwise direction. Since G is inter-triangulated, there exists at least one edge of $E_{i,i+1}$ incident on c that is encountered between B and B' when going around c in the clockwise direction. Such an edge of $E_{i,i+1}$ is called a *separating edge* because it separates blocks B and B' around c. For example, in Figure 1(a), vertex 8 is a cut-vertex of G_1 and edge $(8, g)$ is a separating edge that separates blocks I and B. G is said to be *canonical* if it is inter-triangulated and for any i ($i = 0, \ldots, k-2$) and for any two clockwise consecutive blocks B, B' of G_{i+1} around a cut-vertex, there is exactly one separating edge.

We describe now how a given 2-outerplane graph G can be augmented to become canonical without changing the levels of its vertices. This augmentation technique will be used as a basic step to augment a k-outerplanar graph to become canonical. The augmentation technique consists of the following steps.

1. For each internal face f of the graph G_0, consider the subgraph of G_1 that lies inside f. If this subgraph is not connected, add edges until it is connected.

2. For each vertex $v \in V_1$, if v has no neighbour in G_0, make v adjacent to a vertex of G_0 in such a way that the graph remains planar. This can be done, for example, as follows. For each mixed face f, choose an arbitrary vertex u of f that is in V_0 and connect it to all the vertices of V_1 that are in f and that have no neighbours in V_0.

3. Triangulate the mixed faces of the graph obtained from previous steps. This can be done, in several ways. For example we can proceed as follows.

   (a) Let f be a mixed face. If f has three consecutive vertices u, c, v such that $u, c, v \in G_1$ and c is a cut-vertex of G_1 then (u, v) is not an edge of G. We add (u, v) inside f, i.e. u, c and v now form a triangle. Repeat this step as often as possible for all mixed faces.

   (b) Let f be a non-triangular mixed face. Let $v_0, v_1, \ldots, v_{a-1}$ ($a \geq 1$) be the vertices of f in V_0 in clockwise order when walking on the external face of G_0. Let $w_0, w_1, \ldots, w_{b-1}$ ($b \geq 1$) be the vertices of f in V_1 in clockwise order when walking on the external face of G_1. If $b = 1$, triangulate by adding edges (w_0, v_1), …, (w_0, v_{a-2}). If $b = 2$, then $a > 1$ because the face is non-triangular. In this case either v_0 and w_1 are not adjacent or v_{a-1} and w_0 are not adjacent. Assume that v_0 and w_1 are not adjacent (the other case is analogous); add edges (w_1, v_0), …, (w_1, v_{a-2}). If $b > 2$ add edges (v_0, w_1), …, (v_0, w_{b-2}) and, if $a > 1$, edges (w_{b-2}, v_1), …, (w_{b-2}, v_{a-1}).

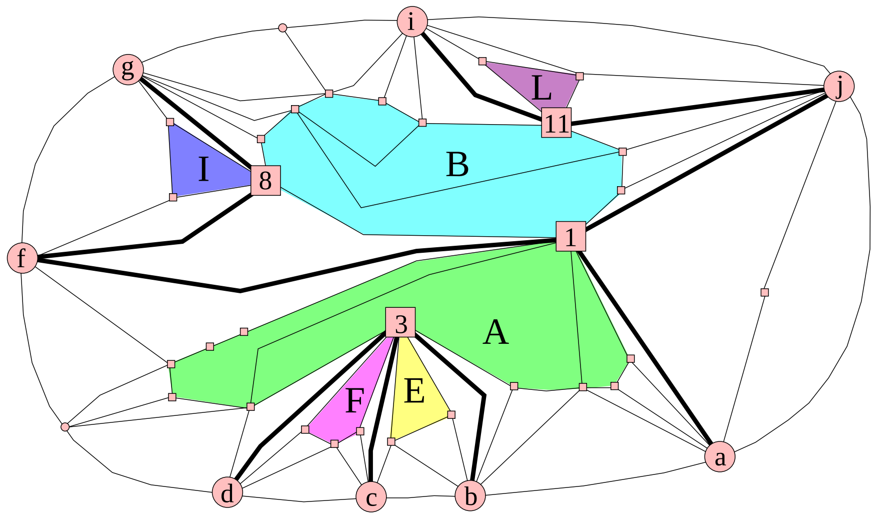
4. Let f be a mixed face of the inter-triangulated graph obtained from previous steps and let x, v, and y be the vertices in the boundary of f. If $x, y \in V_0$ and $v \in V_1$ then add a dummy vertex w inside f and connect it to x, v, and y. Vertex w is assigned to level 1.

Figures 1(a) and 1(b) show a 2-outerplane graph G and the graph G' obtained from the augmentation technique above. Edge $(1, 13)$ is added during Step 1, to make G_1 connected. The edges from vertex f to the vertices of block A are added during Step 2. Edge $(f, 9)$ is added during Step 3. Finally, blocks D, G, H, I, J, K, M, N, O, P, and Q are "created" during Step 4.

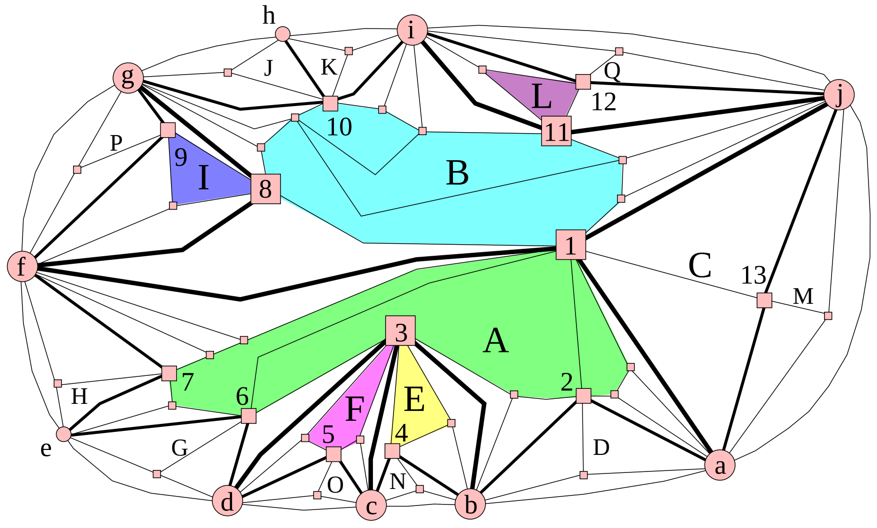
Lemma 1 *Let G be a connected 2-outerplane graph with n vertices. There exists an $O(n)$-time algorithm that computes an augmented graph G' such that* (1) *G' is 2-outerplane,* (2) *the levels of the vertices of G are preserved in G',* (3) *G' is canonical, and* (4) *the number of vertices of G' is $O(n)$.*

Proof: Suppose we use the augmentation technique described above to compute G'. We first prove that G' is 2-outerplane. Since each edge $e = (u, v)$ added to G during Steps 1, 2 and 3 is added inside a face whose boundary contains u and v, these edges do not violate planarity. Step 4 adds a dummy vertex w inside a triangular face f and adds edges from w to the vertices of f. Also these edges do not create crossings. Therefore G' is planar.

Step 1 adds edges connecting vertices in different blocks of G_1; the addition of these edges do not create multiple edges because they connect vertices in different connected components of G_1; also the levels of the vertices is not changed. During Step 2 edges are added between vertices of V_0 and vertices of V_1 and the level of the vertices is not changed. By construction these edges connect non adjacent vertices and thus no multiple edge is created. After the



(a)



(b)

Figure 1: (a) A 2-outerplane graph G. The blocks of G_1 are highlighted and labeled with capital letters. The cut-vertices of G_1 are numbered squares, and their separating edges are bold. (b) The augmented form of G. The new blocks, cut-vertices and separating edges are highlighted.

execution of Step 2, each vertex of V_1 is adjacent to at least a vertex of V_0. This implies that any edge added after Step 2 does not change the level of the vertices, provided that planarity is not violated. Therefore levels are preserved also after the execution of Step 3. Step 4 adds a single vertex w into a mixed triangular face and connects it to a vertex c of G_1 and two vertices of V_0. The dummy vertex w is assigned to level 1, vertex c becomes a cut-vertex of G_1 and its level is not changed. Hence each vertex of G' is assigned either to level 0 or to level 1. It follows that G' is 2-outerplane. Also, the levels of the vertices of G are preserved in G'.

We prove now that Step 3 triangulates all the mixed face of the graph obtained after Step 1 and 2. When Step 3(a) is executed, some cut-vertices c of G_1 are transformed into non-cut-vertices by adding edges between two neighbours u and v of the cut-vertices c. Since c is a cut-vertex then vertices u and v are not adjacent and edge (u, v) can be added without creating multiple edges.

Let f be a non-triangular mixed face of the graph obtained after the execution of Step 3(a). Let $v_0, v_1, \ldots, v_{a-1}$ ($a \geq 1$) be the vertices of f in V_0 in clockwise order when walking on the external face of G_0. Let $w_0, w_1, \ldots, w_{b-1}$ ($b \geq 1$) be the vertices of f in V_1 in clockwise order when walking on the external face of G_1. Step 3(b) considers the following cases:

$b = 1$, i.e. f has only one vertex w_0 of level 1. Since f is not triangular we have $a > 2$, i.e. f has at least three vertices of level 0. Vertex w_0 is not adjacent to vertices $v_1, \ldots, v_{a-2}$: if w_0 were adjacent to these vertices from outside face f, then either v_0 or v_{a-1} would not be on the external face of G_0. Therefore edges (w_0, v_1), …, (w_0, w_{a-2}) split f into $a - 1$ triangular faces without creating multiple edges.

$b = 2$, i.e. f has two vertices w_0 and w_1 of level 1. Since f is not triangular then $a > 2$, i.e. f has at least two vertices v_0 and v_1 of level 0. Vertices w_0 and w_1 are not adjacent to vertices $v_1, \ldots, v_{a-2}$ (if they exist): if w_0 and/or w_1 were adjacent to these vertices from outside face f, then either v_0 or v_{a-1} would not be on the external face of G_0. Also, either v_0 and w_1 are not adjacent or v_{a-1} and w_0 are not adjacent: if both edges (v_0, w_1) and (v_{a-1}, w_0) existed then the subgraph induced by the vertices of f would be a subdivision of K_4 with all its vertices on the same face, which is impossible. Assume that v_0 and w_1 are not adjacent (the other case is analogous). Then, edges (w_1, v_0), …, (w_1, v_{a-2}) split f into a triangular faces and no multiple edge is created.

$b > 2$, i.e. f has more than two vertices w_0 and w_1 of level 1. In this case v_0 is not adjacent to vertices $w_1, \ldots, w_{b-2}$: if v_0 were adjacent to one of these vertices from outside face f, then such vertex would be a cut-vertex of G_1; however, since we execute Step 3(a), no vertex among $w_1, \ldots, w_{b-2}$ can be a cut-vertex. This imply that edges (v_0, w_1), …, (v_0, w_{b-2}) can be added without creating multiple edges. The addition of these edges splits f into $b-2$ triangular faces plus, if $a > 2$, a non triangular face f' whose boundary is $v_0, \ldots, v_{a-1}, w_{b-1}, w_{b-2}$. Vertex w_{b-2} is not adjacent to vertices $v_1, \ldots, v_{a-1}$: if w_{b-2} were adjacent to these vertices from outside face f', then either v_0 or v_{a-1} would not be on the external face of G_0. It follows that edges (w_{b-2}, v_1), …, (w_{b-2}, v_{a-1}) split f' into a triangular face without creating multiple edges.

By the discussion of the cases above we have that G' is inter-triangulated. In order to prove that it is canonical we need to show that there are not two separating edges $e_1 = (c, v_1)$ and $e_2 = (c, v_2)$ between two consecutive blocks around a cut-vertex c. Suppose, as a contradiction, that such two edges exist. Since the graph is inter-triangulated then the two edges belong to the boundary of a triangular face whose vertices are $c \in V_1$ and $v_1, v_2 \in V_0$. But such a face cannot exist because we apply Step 4.

Finally, it is easy to see that the number of vertices of G' is $O(n)$. Namely, at most one vertex is added to each face of the graph obtained after Steps 1,2, and 3. The number of such faces is $O(n)$ since the graph is planar and therefore at most $O(n)$ dummy vertices are added.

Concerning the time complexity, Step 1 computes the connected components of G_1 and adds edges, which can be done in $O(n)$ time. For each face, Step 2 requires time $O(deg(f))$, where $deg(f)$ is the number of edges on the boundary of f. Iterating over all faces we have that this step can be performed in $O(n)$ time. Analogously, Steps 3(a) and 3(b) decompose the mixed faces by adding edges to E_1 and to $\overline{E}_{0,1}$. All the edges can be found in $O(deg(f))$, where $deg(f)$ is the number of edge on the boundary of f. Iterating over all faces we have that these steps can be performed in $O(n)$ time. Finally Step 4 consists of checking the boundary of every face f and possibly placing a new vertex in f. So this step can also be completed in $O(n)$ time. □

Besides being canonical, the augmented graph G' has some additional properties that will be useful in the following sections to ease the description of our drawing algorithm and the proof of its correctness. To state these properties we need some more definitions.

Let G be a 2-outerplane graph and let B be a block of G_1. Let c_0 and c_1 be two (possibly coincident) cut-vertices of G_1 such that, walking clockwise on the external boundary of B: (i) c_0 precedes c_1; (ii) there is no cut-vertex between c_0 and c_1. Let B_0 be the block that precedes B in the clockwise order around c_0 and let B_1 be the block that follows B in the clockwise order around c_1. Let (c_0, v_0) be the separating edge between B_0 and B and let (c_1, v_1) be the separating edge between B and B_1 (note that v_0 and v_1 may coincide if c_0 and c_1 are distinct). Let V_1^p be the set of vertices that are encountered between c_0 and c_1 while walking clockwise on the external boundary of B and let V_0^p be the set of vertices that are encountered between v_0 and v_1 while walking clockwise along G_0, visiting only those vertices of V_0 that are adjacent to vertices in V_1. The subgraph G^p induced by $V_0^p \cup V_1^p$ is called a *portion* of G. Also, we say that G^p is the *portion of* G *defined by* (c_0, v_0) *and* (c_1, v_1). The following properties hold for a 2-outerplanar graph in its augmented form.

Property 1 *Let G be a 2-outerplanar graph and let G' be the graph obtained from the application of the augmentation technique. Each connected component of G'_1 has at least one cut-vertex.*

Proof: Assume, by contradiction, that there exists a connected component K of G'_1 that is biconnected. Since G' is inter-triangulated, there exists a triangular mixed face f consisting of one vertex v of K and two vertices x and y of V_0. But this is impossible because of Step 4. $\Box$

Property 2 *Let G be a 2-outerplanar graph and let G' be the graph obtained from the application of the augmentation technique. For each vertex $u \in V_0$ that is adjacent to at least a vertex of V_1, there is at least one separating edge having u as an end-vertex.*

Proof: Let u be a vertex of V_0 that is adjacent to at least a vertex of V_1 and let (u, v_0), (u, v_1), …, (u, v_{h-1}) ($h \geq 1$)be the edges connecting u to vertices of V_1 in the circular clockwise order around u. Consider vertex v_0 (same argument apply to v_{h-1} if v_0 and v_{h-1} are distinct). Since G' is inter-triangulated v_0 is adjacent to at least another vertex of V_0 distinct from u. Consider two edges $e_1 = (v_0, u_1)$ and $e_2 = (v_0, u_2)$ incident to v_0 such that: (i) $u_1, u_2 \in V_0$, (ii) there is no other edge connecting v_0 to a vertex of V_0 between them in the clockwise order around v_0. Because of Step 4, there must exists a block of G'_1 ("real" or dummy) that is inside the cycle defined by v_0,u_1 and u_2. This implies that v_0 is a cut-vertex of G' and that edge (v_0, u) is a separating edge. $\Box$

Property 3 *Let G be a 2-outerplanar graph and let G' be the graph obtained from the application of the augmentation technique. Let G^p be a portion of G' defined by (c_0, v_0) and (c_1, v_1), with $c_0 \neq c_1$. Then $v_0 = v_1$.*

Proof: Suppose $v_0 \neq v_1$. Let V_0^p be the vertices of G^p that are vertices of V_0 and let V_1^p be the vertices of G^p that are vertices of V_1. Let B the block of G_1 that contains the vertices of V_1^p. Since $c_0 \neq c_1$, B is a "real" block, i.e. it is not a block added by Step 4 of the augmentation technique, because the dummy blocks created by the augmentation have only one cut-vertex. It follows that no vertex of V_1^p is a dummy vertex. If V_1^p only consists of the two vertices c_0 and c_1, we have $v_0 = v_1$ since G' is inter-triangulated. Suppose there exists a vertex $v \in V_1^p$ with $v \neq c_0$ and $v \neq c_1$ that is adjacent to more than one vertex of V_0. Consider two edges $e_1 = (v, u_1)$ and $e_2 = (v, u_2)$ incident to v such that: (i) $u_1, u_2 \in V_0$, (ii) there is no other edge connecting v to a vertex of V_0 between them in the clockwise order around v. Because of Step 4, there must exists a block of G'_1 ("real" or dummy) that is inside the cycle defined by v,u_1 and u_2. This implies that v is a cut-vertex of G'. But this is impossible since by definition of portion there is no cut-vertex encountered between c_0 and c_1 while walking clockwise on the boundary of G_1. Therefore any vertex $v \in V_1^p$ with $v \neq c_0$ and $v \neq c_1$ is adjacent to one vertex of V_0 from which we derive that $v_0 = v_1$ since G' is inter-triangulated. $\Box$

Property 4 *Let G be a 2-outerplanar graph and let G' be the graph obtained from the application of the augmentation technique. Let K be a connected component of G_1 and let B be a block of K. If B has only one cut-vertex, then the block B is a dummy edge (c, v) and v is a dummy vertex.*

Proof: Assume that B has only one cut-vertex but B is not a dummy block, i.e. it is not a block added by Step 4 of the augmentation technique. Since G' is canonical, there exists a (non-dummy) vertex w of B different from c that is adjacent to more than one vertex of V_0. Consider two edges $e_1 = (w, u_1)$ and $e_2 = (w, u_2)$ incident to w such that: (i) $u_1, u_2 \in V_0$, (ii) there is no other edge connecting w to a vertex of V_0 between them in the clockwise order around w. Because of Step 4, there must exists a block of G'_1 ("real" or dummy) that is inside the cycle defined by w,u_1 and u_2. This imply that w is a cut-vertex of G'. But this is impossible since B has only one cut-vertex. □

The augmentation technique described can be used as a basic step in order to augment a k-outerplanar graph G to a k-outerplanar canonical graph G'. More precisely the following lemma holds.

Lemma 2 *Let G be k-outerplane graph with n vertices. There exists an $O(n)$-time algorithm that computes an augmented graph G' such that:* (1) *G' is k-outerplane,* (2) *the levels of the vertices of G are preserved in G',* (3) *G' is canonical, and* (4) *the number of vertices of G' is $O(n)$.*

Proof: The augmentation technique explained above can be repeated $k - 1$ times. More precisely, for each $i = k - 2, k - 1, \ldots, 1, 0$ the augmentation technique can be applied to $G_{i,i+1}$. By Lemma 1 each iteration produces a 2-outerplane graph $G'_{i,i+1}$ in canonical form, so the augmentation produces a k-outerplane graph G'. Since each vertex of G belongs to at most two graphs $G_{i,i+1}$, it follows that G' is constructed in $O(n)$ time. □

Corollary 1 *If the augmented graph G' from Lemma* 2 *has a k-radial drawing, then G has a k-radial drawing.*

5 Equipped BC-trees

We now introduce the equipped BC-tree data structure, which is an extension of the well-known block cut-vertex tree [15]. Let G be a 2-outerplane graph. Based on Corollary 1 we assume that the augmentation technique described in Section 4 has been applied to G and that, therefore, G is canonical and Properties 1, 2, 3 and 4 hold for G. Let K be a connected component of G_1. Recall that by Property 1, K has at least one cut-vertex. An *equipped BC-tree* T of K is a rooted tree such that:

- T has three types of nodes:
  - For each block (i.e. biconnected component) B of K, T has a B-*node* β_B.

  - For each cut-vertex c of K, T has a *C-node* γ_c.
  - For each separating edge e of G that is incident to a cut-vertex of K, T has a *D-node* δ_e.

- The edges of T are of two types:
  - For each cut-vertex c and for each block B that contains c, T has an edge (γ_c, β_B).
  - For each cut-vertex c and for each separating edge e incident to c, T has an edge (γ_c, δ_e).

- The order of the children of each node of T reflects the planar embedding of G: let $B_0, e_0, B_1, e_1, \ldots, B_h, e_h$ $(h \geq 1)$ be the sequence of blocks and separating edges incident to c in clockwise order. Then β_{B_0}, δ_{e_0}, β_{B_1}, δ_{e_1}, …, β_{B_h}, δ_{e_h} are incident to γ_c in this clockwise order.

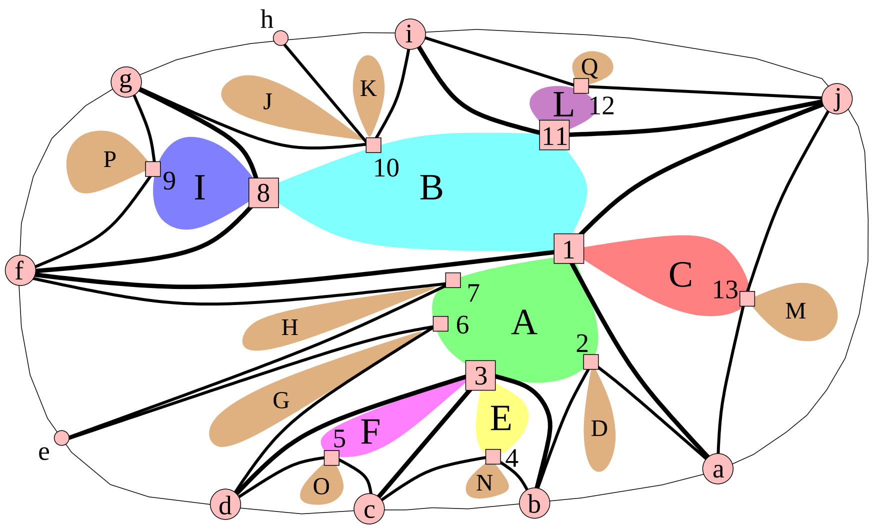
- The root of T is an arbitrary C-node.

The equipped BC-tree of the graph of Figure 1(b) is shown in Figure 2(b). The tree is represented with the root in the bottom of the picture and the leaves in the top so that the left-to-right order in the picture reflects the clockwise order of the nodes in the tree.

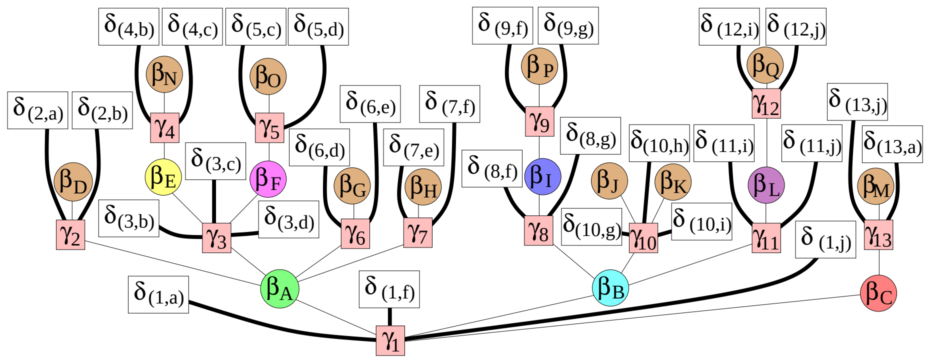
Since G is canonical, a C-node γ_c of T cannot have two adjacent B-nodes that are consecutive in clockwise ordering around γ_c; also, it cannot have two adjacent D-nodes that are consecutive in clockwise ordering. Hence, if γ_c is not the root of T, its parent is a B-node and its leftmost and rightmost children are D-nodes. If γ_c is the root of T we arbitrarily choose a D-node as its leftmost child; as a consequence the rightmost child is a B-node. See for example Figure 2(b) where the rightmost child of the root is the B-node β_C, while the leftmost child is the D-node $\delta_{(1,a)}$. The following lemma can be proved by using standard techniques for BC-trees [14].

Lemma 3 *Let G be a canonical 2-outerplane graph with n vertices. There exists an $O(n)$-time algorithm that computes an equipped BC-tree for each connected component of G_1.*

Let c and (c, v) be a cut-vertex and a separating edge of K, respectively. Vertex v is called a *separating vertex* of γ_c. For example, in Figure 2(b), vertices b, c, d are separating vertices of the C-node γ_3. Denote by β_{B_0}, δ_{e_0}, β_{B_1}, δ_{e_1}, … , β_{B_h}, δ_{e_h} $(h \geq 1)$ the alternate sequence of B- and D-nodes adjacent to γ_c in clockwise order and let $e_i = (c, v_i)$ $(0 \leq i \leq h)$. For each β_{B_i} edges e_{i-1} and e_i are called the *left separating edge* and the *right separating edge* of β_{B_i}, respectively $(1 \leq i \leq h)$. Also the vertices v_{i-1} and v_i are called the *left separating vertex* and the *right separating vertex* of β_{B_i}, respectively $(1 \leq i \leq h)$. A *separating edge* (*separating vertex*) of β_{B_i} is either its left or right separating edge (vertex). If γ_c is not the root of T, let β_{B_0} be the parent of γ_c. Vertices v_0 and v_h are called the *leftmost separating vertex* and the *rightmost separating vertex* of γ_c, respectively.



(a)



(b)

Figure 2: (a) A schematic representation of the structure of the graph of Figure 1(b). The skeleton is highlighted with thick edges. (b) An equipped BC-tree of the graph of Figure 1(b) rooted at γ_1.

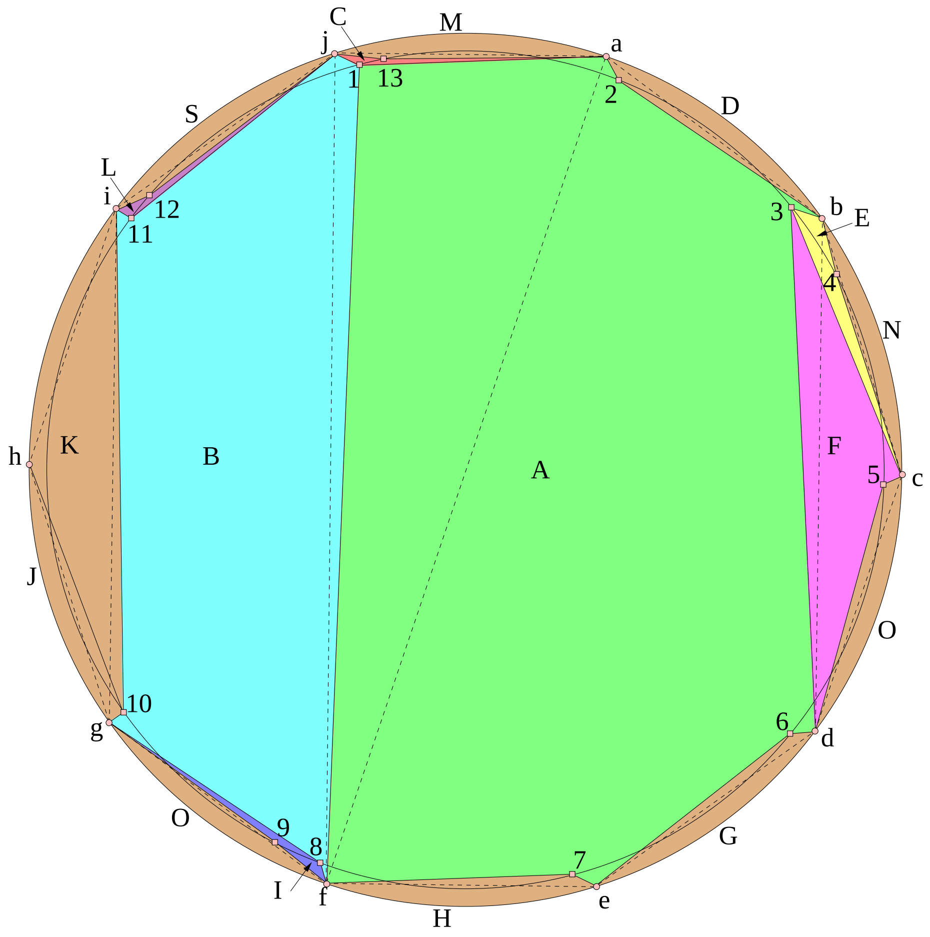


Figure 3: A drawing of the skeleton of the graph of Figure 1(b). The labels of the regions reflect those of the corresponding blocks.

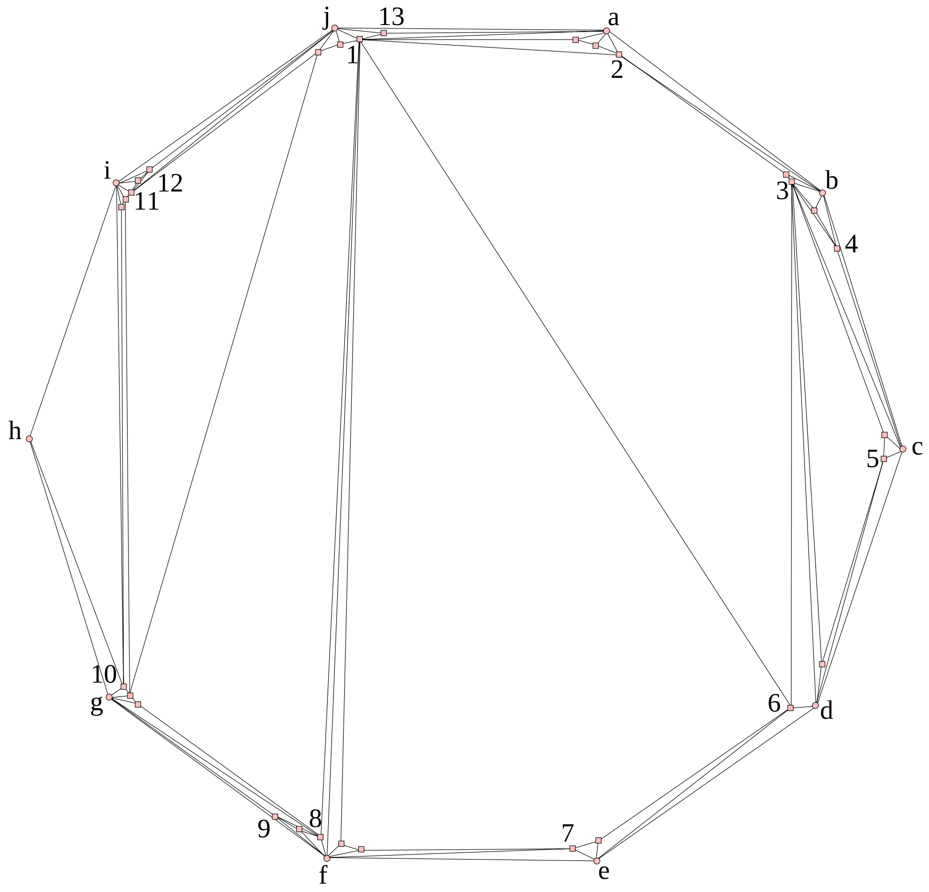


Figure 4: A level-preserving 2-radial drawing of the graph of Figure 1(a).

6 Radial Drawings of 2-outerplanar Graphs

Let G be a 2-outerplane graph. In this section we show how to compute a 2-radial drawing of G. This result will be a basic building block for the drawing techniques and the characterization in Section 7. As explained in Section 5 we assume that the augmentation technique of Section 4 has been applied to G and that, therefore, G is canonical and Properties 1, 2, 3 and 4 hold for G.

Let K be a connected component of G_1. The subgraph of G induced by the separating edges of K is called the *skeleton* of K and is denoted as $skel(K)$. The *skeleton* of G is the union of all $skel(K)$, for every connected component K of G_1. We denote it by $skel(G)$. For example in Figure 1(a), 1(b), and 2(a) the bold edges highlight the skeleton of the graph.

In order to use the algorithm as the basic tool to compute a k-radial drawing of a graph with outerplanarity k (see Section 3), we assume that it receives as input a drawing Γ_0 of G_0 on a circle C_0 and that it computes a drawing Γ of G without changing Γ_0, i.e. $\Gamma_0 \subset \Gamma$. The only hypothesis about Γ_0 is that it preserves the planar embedding of G_0. The algorithm consists of three main steps:

1. **Choice of C_1**: The radius of circle C_1 is determined.

2. **Drawing the skeleton of G**: the drawing of $skel(G)$ is computed, i.e. the cut-vertices of G_1 are placed on C_1 and the separating edges are drawn. The drawing of the skeleton divides the interior of the two circles into regions, one for each block of G_1. Figure 3 shows a drawing of the skeleton of the graph of Figure 1(b).

3. **Drawing the remaining vertices**: The drawing of G is completed by defining the coordinates of the vertices of G_1 that are not cut-vertices. Figure 4 shows a level-preserving 2-radial drawing of the graph of Figure 1(a).

A more detailed description of the three steps of the algorithm follows.

6.1 Choice of C_1

Let K be any connected component of G_1. The radius r_1 of C_1 depends on the drawing of G_0. As we explained in the high-level description of the algorithm given above, the drawing of the skeleton divides the interior of the two circles into regions, one for each block of G_1. Radius r_1 is chosen so that the region associated with each block of K contains an arc of C_1. This will be necessary in order to draw the vertices of the blocks inside their associated regions.

Let $K_0, K_1, \ldots, K_h$ be the connected components of G_1 and let T_j be the equipped BC-tree of K_j ($j = 0, \ldots, h$). For each B-node of T_j with separating vertices v_l, v_r, compute the distance between the point representing v_l and the point representing v_r in Γ_0. Let d_j be the minimum of these distances over all B-nodes of T_j and let $d = \min\{d_j \mid j = 0, \ldots, h\}$. We define the radius r_1 of C_1

to be such that C_1 intersects the chords of C_0 with length d, i.e. $r_1 > \sqrt{r_0^2 - \frac{d^2}{4}}$, where r_0 is the radius of C_0. Computing the radius of C_1 can be performed in a time that is linear in the number of blocks of G, and therefore linear in the number of vertices of G, since the graph is planar.

6.2 Drawing the skeleton of G

Given any circle C, and two points a and b on C, the arc of C traversed when moving from a to b clockwise will be denoted as $\langle a, b\rangle$. Points a and b will be called the *first point* and the *last point* of the arc, respectively. Each point of the arc distinct from a and b will be referred to as an internal point of $\langle a, b\rangle$. Let p be a point of C and let q be a point outside C. Point p is *visible* from q if the segment $\overline{pq}$ does not cross C. The set of points of C that are visible from q is an arc called the *visible region of q on C*. Note that, the first and the last points of the visible region of q on C are the intersection points between C and the straight lines through q tangent to C. The *intersection point of $\overline{pq}$ on C* is either p, if p is visible from q, or the crossing between $\overline{pq}$ and C, otherwise. Let Γ be a 2-radial drawing of G on two circles C_0 and C_1 and let p be a point on C_1. A *free arc* of p is a maximal arc of C_1 having p as one end-point and containing neither vertices of Γ nor crossings between an edge of Γ and C_1. Point p has always two free arcs, one moving from p clockwise: the *right free arc of p*, and the other moving from p counterclockwise: the *left free arc of p*. Let B be a block of a connected component K of G_1 and let γ_c be the parent of β_B in the equipped BC-tree T of K. Let v_l and v_r be the left separating edge and the right separating edge of β_B, respectively. The angle measured going clockwise from $\overline{cv_l}$ to $\overline{cv_r}$ in Γ is called the *corner* of B or also the *corner* of β_B.

A drawing of $skel(G)$ can be computed by drawing independently the skeleton of each connected component of G_1. Namely, the algorithm computes a drawing of $skel(K)$ inside the polygon of Γ_0 representing the face of G_0 that contains K in G. Since G is canonical two different connected components of G_1 are inside different faces of G_0 and therefore their skeletons are drawn inside different disjoint polygons. We want to compute a drawing of K such that the corner of each block of K is convex. This property is important for proving the planarity of the drawing of $skel(G)$. The algorithm performs a BFS visit of T starting from the root and visiting the children of each node according to their clockwise order. We distinguish between the case when it is possible to choose the root of T as a C-node with at least three B-children and the case when this is not possible.

The root of T has at least three B-children. When a C-node γ_c is visited, c is drawn on C_1 together with all its incident separating edges. Two cases are possible:

- γ_c *is the root of* T. Let $\mathcal{P}$ be the convex hull of the separating vertices of γ_c. From the choice of C_1 (Subsection 6.1), every side of $\mathcal{P}$ crosses C_1 in two distinct points. This implies that the interior of $\mathcal{P}$ contains a set of arcs of C_1. Draw c as a point of one of these arcs. See for example the cut-vertex 1 in Figure 3.

- γ_c *is not the root of* T. Let u_l and u_r be the leftmost and the rightmost separating vertices of γ_c, respectively. By the choice of C_1 (Subsection 6.1), $\overline{u_l u_r}$ crosses circle C_1 in two distinct points; let p be the intersection point that is closer to u_l, and denote by α the intersection of the left free arc of p with the visible region of u_l. Vertex c is drawn as a point of α.

The root of T has exactly two B-children. When the root has only two B-children, we cannot guarantee that the corners of all the B-nodes are convex. Namely if the root γ_c of T has only two B-children, then it has also only two D-children, i.e. c has only two separating edges incident to it. Thus, if we guarantee that one of the angles between the two segments that represent these two edges is less than $180°$ the other angle between the same two segments is greater than $180°$. We now describe how to deal with this case.

If there does not exist a cut-vertex shared by at least three blocks, then there exists at least a block B_r whose vertices are adjacent to more than two vertices of G_0. Namely, if such a block did not exist then there would be a face of G_0 of degree two, which is impossible since G does not have multiple edges.

Let T be the equipped BC-tree of K. We choose the root γ_{c_r} of T to be a C-node associated with a cut-vertex c_r of B_r. As in the case when the root has more than two B-children we can arbitrarily choose the rightmost (or equivalently the leftmost) child of γ_{c_r}. In this case we choose β_{B_r} as the rightmost child of γ_{c_r}.

The cut-vertex c_r is placed on C_1 similarly to the non-root cut-vertices in the case when the root has at least three B-children. Namely, let u_l and u_r be the left and the right separating vertices of the B-child β_B of γ_{c_r} different from β_{B_r}. Let p be the intersection point between $\overline{u_l u_r}$ and C_1 that is closer to u_l. Place c_r in the intersection between the left free arc of p and the visible region of u_l. After the placement of c_r the other cut-vertices of β_{B_r} are placed on C_1 according to the clockwise order around the boundary of B_r. For each cut-vertex c, let u_l and u_r be the leftmost and the rightmost separating vertices of γ_c. Let p be the intersection point between $\overline{u_l u_r}$ and C_1 that is closer to u_l. Place c in the intersection between the left free arc of p and the visible region of u_l.

After the placement of all the cut-vertices of B_r all the other cut-vertices of G_1 are placed with the same technique described for the case when the root has at least three B-children.

Lemma 4 *Let G be a 2-outerplane graph with n vertices. Let Γ_0 be an embedding preserving 1-radial drawing of G_0. There exists an $O(n)$-time algorithm* *that computes an embedding preserving* 2*-radial drawing* Γ *of* $G_0 \cup skel(G)$ *such that* $\Gamma_0 \subset \Gamma$*. Also,* Γ *is a level-preserving drawing.*

Proof: We consider first the case when the root has at least three B-children and prove that the algorithm described above computes a drawing for which the statement holds. First we observe that for each cut-vertex c of G_1 the arc α of C_1 where c has to be placed has non-zero length. Point p is in the visible region of u_l since the segment $\overline{u_l p}$ does not cross C_1 and therefore the arc α has non-zero length. This proves that our algorithm always finds a point to represent the cut-vertex c.

In order to prove that Γ is planar we concentrate first on a single connected component K of G_1 with equipped BC-tree T. We prove the following claim.

Claim *When a cut-vertex* c *is drawn on* C_1 *the separating edges incident to* c *can be drawn without crossing any other existing edge. Also, the corner of each* B*-child of* γ_c *is convex.*

Let $c_0, c_1, \ldots, c_k$ be the cut-vertices of K in the order they are added to the drawing. The proof is by induction on k.

If $k = 0$, then c_0 is the root of T and it is drawn on C_1 as a point inside the polygon $\mathcal{P}$ defined by the convex hull of the separating vertices of γ_c (note that γ_c has at least three D-children and therefore $\mathcal{P}$ is indeed a polygon). Polygon $\mathcal{P}$ is convex and for each separating edge (c_0, v) incident to c_0, v is a vertex of $\mathcal{P}$. Therefore these edges can be drawn without crossing each other and without cross any other existing edge. Also, the corner of each B-child of γ_c is the angle between two consecutive separating edges incident to c_0; for any pair of consecutive separating edges incident to c_0 the angle between them is clearly convex. Then the statement holds for c_0.

Suppose now that the claim holds for $k-1$, i.e. the first $k-1$ vertices have already been placed so that the claim holds for them. Let β_B be the parent of γ_{c_k} and let γ_{c_j} ($j < k$) be the parent of β_B. Let v_l and v_r be the left separating edge and the right separating edge of β_B, respectively and let u_l and u_r be the leftmost and the rightmost separating edges of γ_{c_k}, respectively. Denote by $\mathcal{R}$ the region bounded by $\overline{c_j v_l}$, $\overline{c_j v_r}$ and by the polyline representing the path π from v_l to v_r on the boundary of the face f of G_0 that contains K. Since Γ_0 preserves the embedding we have that v_l, u_l, u_r, and v_r appear in this order when walking clockwise on C_0 (possibly with $u_l = v_l$ and with $u_r = v_r$). Therefore $\overline{u_l u_r}$ is contained in $\mathcal{R}$ which is convex by induction. It follows that the point p (i.e. the crossing point between $\overline{u_l u_r}$ and C_1 that is closer to u_l) and the arc α are inside $\mathcal{R}$ and therefore c_k is drawn inside $\mathcal{R}$. For each separating edge (c_k, v) incident to c_k, v is a vertex of π and since $\mathcal{R}$ is convex all these edges can be added to drawing without creating a crossing. Also the angle measured going clockwise from $\overline{c_k u_l}$ to $\overline{c_k u_r}$ is convex by construction and therefore the corner of each B-child of γ_c is convex.

From the proof of the claim we have that the skeleton of a single connected component K of G_1 is drawn completely inside the polygon $\mathcal{P}$ which is contained in the polygon representing the face f. Since G is canonical, then two different connected components of G_1 are inside different faces of G_0 and therefore their skeletons are drawn inside different disjoint polygons. This implies that Γ is planar. Also Γ is embedding preserving and level preserving by construction.

We consider now the case when the root has exactly two B-children. The drawing of all the cut-vertices of B_r along with the separating edges incident to them is crossing-free. Namely, the separating vertices of the cut-vertices of B_r are more than two because the vertices of B_r are adjacent to more than two vertices of G_0 and, by Property 2, if a vertex of V_0 is adjacent to a vertex of V_1 then there is a separating edge incident to it. Hence the convex hull of such vertices is a polygon $\mathcal{P}'$. Also, by Property 3 each cut-vertex is adjacent to two consecutive vertices of $\mathcal{P}'$. Since the placement of the cut-vertices of B_r preserves the embedding, no crossing is possible. Also, for each cut-vertex c of B_r, the corner of each B-node that is a child of a γ_c is convex. Therefore, all the remaining cut-vertices of K can be placed on C_1 according to the technique described for the case when the root has at least three B-children.

Concerning the time complexity of the algorithm, we prove that for each cut-vertex c of G_1 we can choose the arc where c has to be drawn in $O(1)$ time. Consider first the case when the root has at least three B-children. If γ_c is the root of T and it has at least three B-children then, in $O(1)$ time, we can retrieve the coordinates of any three consecutive separating vertices u, v and w of γ_c. The intersection p between $\overline{uv}$ and C_1 that is closer to v and the intersection q between $\overline{vw}$ and C_1 that is closer to v can also be computed in $O(1)$ time. Arc $\langle p, q\rangle$ is one of the arcs of C_1 that are inside polygon $\mathcal{P}$, i.e. one of the arcs where c can be placed.

If γ_c is not the root of T, let u_l and u_r be the leftmost and the rightmost separating vertices of γ_c, respectively. The visible region of u_l can be computed in $O(1)$ time by simple trigonometry. The crossing p between segment $\overline{u_l u_r}$ and C_1 that is closer to u_l can be computed in $O(1)$ time. By Property 3, the first point p_f of the left free arc of p is the intersection point of $\overline{c' u_l}$ on C_1, where c' is the cut-vertex that is encountered before c in the clockwise order around the block B that contains c and c'. The coordinates of c' can be retrieved in $O(1)$ time because either $\gamma_{c'}$ is the grandparent of γ_c or they are consecutive children of β_B. Therefore p_f and the arc α where c has to be placed can be computed in $O(1)$ time.

Consider now the case when the root of T has exactly two B-children and consider the placement of c_r. Also in this case the visible region of u_l and the crossing p between segment $\overline{u_l u_r}$ and C_1 that is closer to u_l can be computed in $O(1)$ time. The first point p_f of the left free arc of p is the crossing between a segment $\overline{u_l v}$ and C_1, where $v \in V_0$ and (u_l, v) is the first edge after those of $\overline{E}_{0,1}$ in the clockwise order around v. Therefore p_f can be computed in $O(1)$ time. It follows that the arc where c_r has to be placed can be computed in $O(1)$ time.

Concerning the other cut-vertices of T, by the same argument as the one for the non-root nodes in the case of the root with at least three children, we can prove that they can be placed in $O(1)$ time. Hence, the overall time complexity is $O(n)$. □

6.3 Drawing the remaining vertices

We complete the drawing by drawing independently each portion of G. Let G^p be a portion of G defined by (c_0, v_0) and (c_1, v_1).

If $c_0 = c_1$ then G^p has only one cut-vertex and by Property 4, we have that B is a dummy edge (c_0, v) and v is a dummy vertex. Hence we do not need to draw v.

If $c_0 \neq c_1$, by Property 3 we have that $v_0 = v_1$. Let q_0 and q_1 be the intersection points of $\overline{c_0v_0}$ and $\overline{c_1v_1}$ on C_1, respectively. In this case the polygon defined by v_0, q_0 and q_1 is a triangle. All the vertices of G^p that are in V_1 are adjacent to v_0 and they are drawn in the arc $\langle q_0, q_1\rangle$ according to their clockwise order around the boundary of the block B to which they belong. It is immediate to see that the drawing computed in this way is crossing-free, level preserving and embedding preserving. Also the drawing of each portion G^p can be computed in $O(n_p)$ time, where n_p is the number of vertices of G^p. We have the following lemma.

Lemma 5 *Let G be a 2-outerplane graph with n vertices. Let Γ_0 be an embedding preserving 1-radial drawing of G_0. There exists an $O(n)$-time algorithm that computes an embedding preserving 2-radial drawing Γ of G such that $\Gamma_0 \subset \Gamma$ and Γ is level-preserving.*

The following theorem summarizes the results of this section.

Theorem 1 *Let G be a 2-outerplane graph with n vertices. G admits a level-preserving 2-radial drawing that preserves the embedding of G. Also there exists an $O(n)$-time algorithm that computes such a drawing.*

7 Computing Minimum Radial Drawings of Planar Graphs

In this section we first characterize the family of graphs that admit a radial drawing on at most k concentric circles and then use the characterization to solve the problem of computing a minimum radial drawing of a planar graph in polynomial time. We start by extending the result of Theorem 1 to k-outerplane graphs.

Theorem 2 *Let G be a graph with outerplanarity k. Then G admits a proper k-radial drawing. Also, given a k-outerplanar embedding of G, there exists an $O(n)$-time algorithm that computes such a drawing, where n is the number of vertices of G.*

Proof: Since G has outerplanarity k then it has a k-outerplanar embedding. We show how to compute a level-preserving k-radial drawing Γ of G that preserves this embedding. This implies that Γ is proper. An algorithm to compute Γ is based on first drawing the subgraph induced by the vertices of level 0 on circle C_0 and then by adding at each step the vertices of level i on circle C_i $(i = 1, \cdots k-1)$. By Lemma 2 we can assume that G is canonical an that for each $G_{i,i+1}$ Properties 1, 2, 3 and 4 hold. Therefore at Step i the subgraph $G_{i-1,i}$ can be drawn by using the algorithm described in Section 6. Since $G = \bigcup_{i=0}^{k-2} G_{i,i+1}$ the computed drawing is a radial drawing of G. We prove that no two edges cross by induction. The drawing of $G_{0,1}$ is planar by Lemma 5. Assume that the drawing of $G_0 \cup \ldots \cup G_{i-1}$ $(i > 1)$ is planar. This implies that each connected component of G_{i-1} is drawn inside a face of $G_0 \cup \ldots \cup G_{i-2}$. By Lemma 5, the drawing of $G_{i-1,i}$ is planar, which implies that the drawing of each connected component K of G_i is drawn inside a face of the drawing of G_{i-1}, and then, from the above argument, the drawing of K is also inside a face of the drawing of $G_0 \cup \ldots \cup G_{i-1}$. This implies that the drawing of $G_0 \cup \ldots \cup G_{i-1} \cup G_i$ is still planar. Therefore, the above described algorithm computes a level-preserving k-radial drawing of G. As for the time complexity, it follows from Lemma 5 that the computation of drawing $G_{i-1,i}$ requires $O(n_i)$ time where n_i is the number of vertices in $G_{i-1,i}$. Therefore the overall time complexity is $O(n)$. □

While the algorithm in Theorem 2 has as input a k-outerplane graph G and computes a drawing of G that satisfies radial properties, the following lemma assumes that a radial drawing is given and studies the combinatorial properties of the represented graph.

Lemma 6 *Let Γ be a k-radial drawing of a graph G. Then G has outerplanarity at most k.*

Proof: We prove the statement by induction on k. If $k = 1$ then Γ is a straight-line drawing with all vertices on a circle, and therefore G is 1-outerplane.

Suppose that the statement is true for $1 \leq j \leq k-1$, and let Γ be a drawing of G on k circles. All the vertices drawn on C_0 are vertices of the external face of G because the drawing is straight-line and crossing-free. Hence, if we remove all the vertices of the external face of G, all vertices drawn on C_0 are removed and we are left with a $(k-1)$-radial drawing. By induction the graph obtained by removing the vertices of the external face of G has outerplanarity at most $(k-1)$ and therefore G has outerplanarity at most k. □

Lemma 6 and Theorem 2 imply the following characterization.

Theorem 3 *Let G be a planar graph. G admits a radial drawing on at most k circles if and only if the outerplanarity of G is at most k.*

Proof: Assume that G has a k-radial drawing. Then, by Lemma 6, G has outerplanarity at most k. Conversely, if G has outerplanarity $j \leq k$, by Theorem 2 it admits a j-radial drawing with $j \leq k$. □

Based on the result of Theorem 3, we can show that the problem of computing a minimum radial drawing of a planar graph G can be solved in polynomial time. The minimum radial drawing has the additional property of being proper, that is an edge always connects either co-circular vertices or vertices on consecutive levels.

Theorem 4 *Let G be a planar graph with n vertices. There exists an $O(n^5 \log n)$-time algorithm that computes a minimum radial drawing of G. Furthermore the computed drawing is proper.*

Proof: Bienstock and Monma [4] describe an algorithm to compute the outerplanarity k of G and to determine a k-outerplanar embedding of G. This algorithm takes $O(n^5 \log n)$ time. The result in [4] together with Theorem 3 imply that k is the minimum number of circles for which there exists a radial drawing of G. The fact that such a drawing is proper is a consequence of Theorem 2. Again by Theorem 2 it follows that the time complexity of the whole algorithm is dominated by the technique in [4]. $\square$

8 Conclusions and Open Problems

The problem studied in this paper can be seen as a partitioning problem: We want to partition the vertices of a planar graph G into the minimum number of levels so that there exists a level-preserving radial drawing of G. Using a result of Bienstock and Monma [4], we proved that there exists a polynomial time algorithm that finds such a partition and computes a corresponding minimum radial drawing. The results in this paper suggest several open problems. We conclude by listing three of those that in our opinion are among the most interesting.

- Compute a minimum radial drawing of a planar graph with n vertices in $o(n^5 \log n)$ time.
- Study the trade-off between minimizing the number of circles and maximizing the angular resolution.
- Our characterization result assumes that the radius of the circles in the radial drawing is not fixed. It would be interesting to study the minimum radial drawability problem in the case of fixed radius.

Acknowledgments

The authors wish to thank the anonymous referees for several useful suggestions.

References

[1] C. Bachmaier, F. J. Brandenburg, and M. Forster. Radial level planarity testing and embedding in linear time. In G. Liotta, editor, *Graph Drawing (GD 2003)*, volume 2912 of *Lecture Notes in Computer Science*, pages 393–405. Springer, 2004.

[2] C. Bachmaier, F. J. Brandenburg, and M. Forster. Track planarity testing and embedding. In P. Van Embde Boas, J. Pokorny, M. Bielikova, and J. Stuller, editors, *SOFSEM 2004: Theory and Practice of Computer Science*, volume 2, pages 3–17, 2004.

[3] T. C. Biedl. Drawing planar partitions I: LL-drawings and LH-drawings. In *Proceedings of the ACM Symposium on Computational Geometry (SoCG'98)*, pages 287–296, 1998.

[4] D. Bienstock and C. L. Monma. On the complexity of embedding planar graphs to minimize certain distance measures. *Algorithmica*, 5(1):93–109, 1990.

[5] S. Bornholdt and H. Schuster, editors. *Handbook of Graphs and Networks: From the Genome to the Internet*. Wiley-VCH, 2003.

[6] S. Card, J. Mackinlay, and B. Shneiderman. *Information Visualization*. Morgan Kaufmann, 1999.

[7] S. Cornelsen, T. Schank, and D. Wagner. Drawing graphs on two and three lines. In S. G. Kobourov and M. T. Goodrich, editors, *Graph Drawing (GD 2002)*, volume 2528 of *Lecture Notes in Computer Science*, pages 31–41. Springer, 2002.

[8] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing*. Prentice Hall, 1999.

[9] E. Di Giacomo and W. Didimo. Straight-line drawings of 2-outerplanar graphs on two curves. In G. Liotta, editor, *Graph Drawing (GD 2003)*, volume 2912 of *Lecture Notes in Computer Science*, pages 419–424. Springer, 2004.

[10] E. Di Giacomo, W. Didimo, G. Liotta, and S. K. Wismath. Curve-constrained drawings of planar graphs. *Computational Geometry: Theory and Applications*, 30:1–23, 2005.

[11] M. Dodge and R. Kitchin. *Atlas of Cyberspace*. Addison Wesley, 2001.

[12] S. N. Dorogstev and J. F. F. Mendes. *Evolution of Networks, From Biological Nets to the Internet and WWW*. Oxford University Press, 2003.

[13] S. Felsner, G. Liotta, and S. K. Wismath. Straight-line drawings on restricted integer grids in two and three dimensions. *Journal of Graph Algorithms and Applications*, 7(4):363–398, 2003.

[14] F. Harary. *Graph Theory.* Addison-Wesley, 1972.

[15] F. Harary and G. Prins. The block-cutpoint-tree of a graph. *Publ. Math Debrecen*, 13:103–107, 1966.

[16] L. S. Heath and A. L. Rosenberg. Laying out graphs using queues. *SIAM Journal on Computing*, 21:927–958, 1992.

[17] M. Juenger and P. Mutzel, editors. *Graph Drawing Software.* Springer-Verlag, 2003.

[18] M. Suderman. Pathwidth and layered drawings of trees. *International Journal of Computational Geometry and Applications*, 14(3):203–225, 2004.

[19] K. Sugiyama. *Graph Drawing and Applications for Software and Knowldege Engineers.* World Scientific, 2002.