Upward Planarity Testing of Outerplanar Dags (Extended Abstract)

Achilleas Papakostas*

Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75083-0688

Abstract. In this paper, we present two polynomial-time algorithms to determine if an outerplanar directed acyclic graph (odag) can be drawn *upward planar*, that is, drawn in planar straight-line fashion so that all arcs point up. The first algorithm checks if the odag has an upward planar drawing that is topologically equivalent to the outerplanar embedding of the odag. This algorithm runs in linear time (which is optimal), and is faster than any previous algorithm known. The second algorithm also checks whether an odag has an upward planar drawing but does not insist that the drawing be topologically equivalent to the outerplanar embedding. This is the first polynomial-time algorithm we know of to solve this problem.

1 Introduction

One can find, in the literature, many algorithms for checking planarity of undirected graphs and for computing planar embeddings of planar graphs. Both planarity checking and planar embedding can be done in linear time [7], [5], [11].

When one is concerned with directed acyclic graphs, the notion of *upward planarity* naturally arises and replaces the notion of planarity for undirected graphs. A directed acyclic graph is *upward planar* if it admits a planar drawing with the additional constraint that all the edges point upwards (i.e., from a lower to a higher y-coordinate). There are many practical situations where upward drawings are required: drawing semantic networks and other knowledge representation diagrams, displaying call graphs in compiler optimization, obtaining E-R diagrams in database design, and any other occasion where a hierarchical structure must be represented.

A combinatorial characterization of the directed acyclic graphs that have an upward drawing is given in [9] and [4] where it is shown that a directed acyclic graph admits an upward drawing if and only if it is a subgraph of a planar st-graph. It has been recently shown that the general problem of testing whether a directed acyclic graph is upward planar is NP-complete [6].

The work described in the present paper is chiefly aimed at obtaining two efficient algorithms for checking upward planarity of outerplanar directed acyclic

* Part of this work was done while the author was at the University of Massachusetts.

graphs. We will see how we can check in linear time (which is optimal) whether an outerplanar embedding is upward. This is our first algorithm and it is described in Sect. 4. In the case that the outerplanar embedding is not upward, we will demonstrate how to obtain a non-outerplanar embedding that is upward planar (assuming one exists) in polynomial time. This is our second algorithm and it is presented in Sect. 5. This algorithm is also the first approach we know of to solve this problem.

Section 2 contains some basic definitions and preliminaries. Section 3 surveys these families of directed acyclic graphs for which determining upward planarity can be done in polynomial time. The paper concludes with some open problems in Sect. 6 .

2 Preliminaries

We assume that the reader is familiar with basic terminology and results of graph theory. Most of the definitions given here are the same as in [1].

A graph G has a *planar* drawing if it has a drawing so that no two edges intersect except, of course, at common endpoints. A graph is *planar* if it admits a planar drawing. A planar graph may have many different planar drawings. Two planar drawings of the same planar graph are *equivalent* if, for each vertex v, they have the same circular clockwise order of the edges incident to v. Two equivalent drawings have the same set of faces too. A planar *embedding* of a planar graph is a collection of planar drawings which are pairwise equivalent.

A graph has an *outerplanar* embedding if it has a planar embedding in which all vertices reside on the same face (called *external* face). Every outerplanar graph has a unique outerplanar embedding.

The *underlying graph* of a directed acyclic graph (or *dag*) is the undirected graph obtained from the dag by considering the edges as undirected. Vertices of a dag which do not have any incoming edges are called *sources*, and vertices that do not have any outgoing edges are called *sinks*. Sources and sinks are also called *switches*.

An *upward drawing* of a dag G is a planar drawing of G with the additional constraint that each edge is a curve rising monotonically in the vertical direction. Given a drawing of a dag, we say that a vertex v is candidate if the outgoing (incoming) edges incident to v appear consecutively around v. If this happens for all vertices of the drawing then the drawing is a *candidate upward drawing*. Every upward drawing of a dag is a candidate upward drawing.

Two upward drawings of the same dag are *equivalent* if their underlying planar drawings are equivalent. A collection of pairwise equivalent upward drawings of a dag is an *upward embedding* of the dag. A planar embedding of a dag in which every vertex is candidate is a *candidate upward embedding* of the dag. Clearly, every upward embedding is a candidate upward embedding.

A dag is *upward planar* if it admits an upward drawing. Candidacy is only a necessary condition for the existence of an upward embedding. Figure 1a shows an upward embedding and Fig. 1b shows a candidate upward embedding which

is not upward. From here on, n will denote the number of vertices in a planar dag.

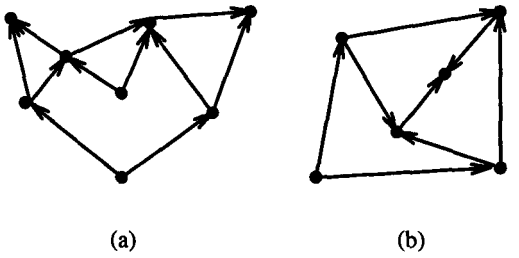

Fig. 1. (a) an upward embedding, (b) a candidate upward embedding which is not upward

3 Upward Planarity of Some Families of Dags

There are algorithms to test upward planarity of some families of dags. One of these families is the family of directed trees for which it is easy to show that they are upward planar.

Di Battista et al [3] showed that a bipartite dag is upward planar iff its underlying graph is planar. Platt [10] showed that single-source and single-sink dags are upward planar iff the underlying graph with a source-to-sink added edge is planar. Hutton and Lubiw [8] developed a $O(n^2)$ algorithm for testing upward planarity of single-source dags. This result was improved by Bertolazzi, Di Battista, Mannino and Tamassia [2] who showed that upward planarity testing of a single-source digraph can be done optimally in $O(n)$ time.

We are going to associate a *capacity* (see [1] for a formal definition of capacity) with every face (internal and external) of a candidate upward embedding. Switches are assigned to faces in whose boundary they appear. A switch can be assigned only to one of the faces that share it, and there cannot be switches that are not assigned to any face.

We call a face *satisfied* if it is assigned precisely as many switches as its capacity. It is proved [1] that:

Theorem 1. *A candidate upward embedding of a dag is upward iff there exists an assignment which satisfies all the faces.*

If we apply this technique for every embedding of a triconnected dag [1] we can check whether it is upward planar in $O(n + r^3 \log r)$ time (r is the number of switches).

An embedded dag whose internal faces are triangles is a *triangular dag*. We can easily show the following proposition:

Proposition 2. *A candidate triangular dag is upward planar iff no internal vertex is a switch.*

A single-source or single-sink outerplanar dag (odag for short) is clearly upward planar. *Circuits* are odags whose underlying graph is a simple cycle. Circuits are known to be upward planar [1]. Also, we can see that the following holds:

Proposition 3. *Every odag whose outerplanar embedding consists of a circuit with a single chord inside it is upward planar.*

Outerplanar dags can be classified into the following three groups:

1. odags whose outerplanar embedding is an upward embedding, and therefore they are upward planar (Fig. 2).
2. odags which are upward planar but their outerplanar embedding is not upward. Figure 3a shows an outerplanar embedding that is not candidate upward (and therefore not upward), and Fig. 3b shows a non outerplanar embedding of the same odag which is upward planar.
3. odags that are not upward planar (Fig. 4).

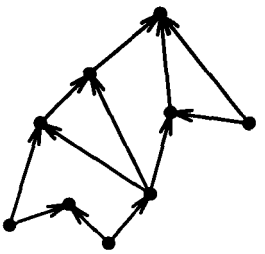

Fig. 2. An outerplanar upward embedding

If we apply Proposition 2 to the case of triangulated outerplanar dags it is clear that every triangulated outerplanar candidate upward embedding is upward planar.

4 Testing Upward Planarity of Outerplanar Embeddings

In this section we will present a linear time algoritm for testing whether an outerplanar candidate upward embedding is upward. Recall that the general way for testing whether a candidate upward embedding is upward (described in [1]) takes $O(n + r^2 \log r)$ time (r is the number of switches). Note that we can easily check in linear time whether an outerplanar embedding is candidate upward.

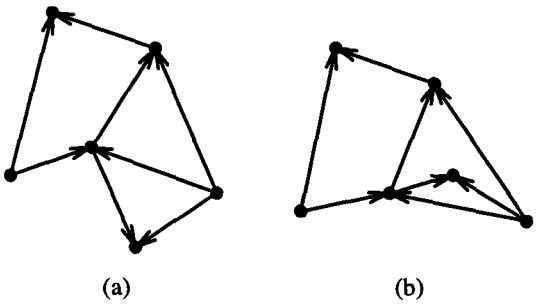

Fig. 3. (a) an outerplanar embedding which is not upward, (b) an upward and non outerplanar embedding of the same odag

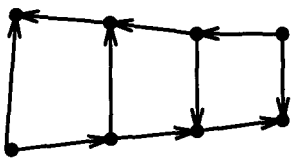

Fig. 4. A non upward planar odag

The dual of an outerplanar embedding is a tree, if we exclude the dual vertex which corresponds to the external face and its incident edges. Our technique is based on a DFS performed on the dual tree and on appropriate coloring of switches of the odag taking place during the search process. The switches are assigned colors during the algorithm as follows:

- *red:* the switch has not been assigned yet to any face, and is thus still available for assignment.
- *brown:* the switch has been assigned permanently to a face, and is thus not available for assignment.

The face of the outerplanar embedding that the DFS is currently visiting is called the *current* face. Only the switches which appear around the current face can be colored at a given time. All switches are originally colored red. The *adjusted* capacity of each internal face is originally equal to its capacity. When we describe the algorithm, we will see how the adjusted capacity is computed.

Figure 5 shows the situation when the current face is c. In the same figure, subgraphs $G_1, G_2, \cdots, G_m$ which appear around c, in clockwise order, have already been visited. When at the current face c, we always try to satisfy its adjusted capacity using the existing red switches around it. We distinguish among the following cases:

1. If the number of red switches around c is equal to the adjusted capacity of c,

then c and all the faces in subgraphs $G_1, G_2, \cdots, G_m$ are satisfied and these red switches become brown.

2. If the number of red switches around c is less than the adjusted capacity of c, then c and $G_1, G_2, \cdots, G_m$ cannot be satisfied so we stop and the embedding is declared not upward.
3. If the number of red switches is more than the adjusted capacity of c then:
   - if neither X nor Y (Fig. 5) is a red switch, then c and $G_1, G_2, \cdots, G_m$ are satisfied (notice that if there are excess red switches then they will be assigned to the external face). All red switches around c become brown.
   - if exactly one of X and Y is a red switch, then it maintains its color (it might be needed later when DFS visits b), whereas all the other red switches around c become brown. c and $G_1, G_2, \cdots, G_m$ are satisfied (the same observation as above holds for excess red switches).
   - if both X and Y are red switches, then we compute the difference between the number of red switches around c and the adjusted capacity of c.
     If the difference is 2 or more, then X and Y maintain their color, the rest of the red switches around c become brown, c and $G_1, G_2, \cdots, G_m$ are satisfied.
     If the difference is 1, then this means that c needs only one of X and Y but we can not currently determine which. So we move one unit of c's adjusted capacity to b, that is the adjusted capacity of b increases by 1. c and $G_1, G_2, \cdots, G_m$ will be satisfied when b is satisfied. X and Y maintain their color and all the other red switches around c become brown.

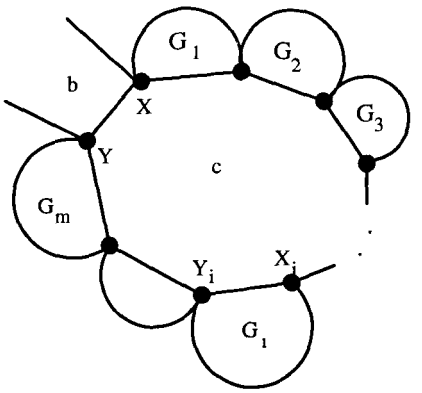

Fig. 5. DFS performed on the dual tree of an outerplanar embedding: c is the current face

Lemma 4. *An outerplanar embedding is upward iff all switches become brown at the end of the DFS.*

The basic observation here is that we always try to satisfy the capacities of the faces with switches that cannot be used by faces which the DFS will visit later. In this way we keep "pushing" as many switches as possible up the tree.

Using appropriate data structures we ensure that the time spent on each current face is order of the number of angles in the interior of the face. Therefore, the algorithm runs in time which is order of the number of angles of the embedding. From the discussion above it follows that:

Theorem 5. *Testing whether an odag has an upward embedding which is topologically equivalent to the outerplanar embedding of the odag can be done in $O(n)$ time.*

5 Folding Faces to Achieve an Upward Embedding

Consider a planar embedding of an outerplanar graph and an edge e of the embedding which does not appear on the boundary of the external face (Fig. 6). Edge e partitions the embedding into two subembeddings A and B on the two different sides of e so that every path from a vertex in A to a vertex in B goes through one of the two vertices to which e is incident.

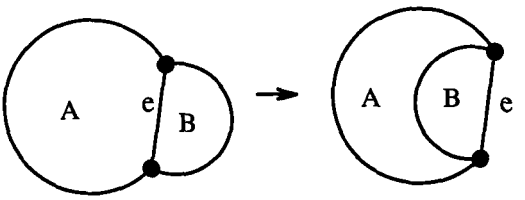

Fig. 6. Folding subgraph B around edge e

A *folding* of B with respect to e is a rotation of subgraph B around e so that B maintains its structure, and is embedded inside that face of A with e on its boundary. This results in a new planar embedding. It is easy to see that for any planar embedding of an outerplanar graph there is a sequence of foldings which will yield this embedding if we start with the outerplanar embedding.

Suppose that an odag does not have an upward embedding which is topologically equivalent to its outerplanar embedding. In that case, we have to check whether there exists a sequence of foldings which, if applied on the outerplanar embedding of the odag, they produce an upward embedding. This is the topic of this section.

A DFS on the dual tree of the outerplanar embedding will be employed again. The current face c along with subgraphs $G_1, G_2, \cdots, G_m$ is called the current *flower* f_c (Fig. 5). Subgraphs $G_1, G_2, \cdots, G_m$ are called the *petals* of f_c. Each one of the G_i's has already been embedded in an upward planar fashion. We will check whether some of the G_i's have to be folded inside c so that the flower f_c has an upward embedding.

Now consider graph $c \cup G_i$. Let (X_i, Y_i) be the edge shared by c and G_i (Fig. 5), and consider all upward embeddings of $c \cup G_i$. In any such embedding, G_i is either outside c, or has been folded inside c, and some *local conditions* hold on c with respect to vertices X_i and Y_i. These local conditions are the different ways that the angles formed by the edges of c on vertices X_i and Y_i are assigned (see [1]) to the internal or external face of c.

We record all valid pairs (i.e., $< G_i$ is inside c, "local condition" on $c >$) that correspond to an upward embedding of $c \cup G_i$. Let L_i be the set of these pairs. We do this for each $i = 1, 2, \cdots, m$. Next, we build the *condition* graph in the following way.

We introduce a vertex for each pair in L_i. We make a directed edge go from vertex $u \in L_i$ to vertex $v \in L_{i+1}$ if all the following conditions hold:

1. there is a vertex which is shared between G_i and G_{i+1}, and let's call this vertex Y_i.
2. the local conditions with respect to Y_i on the two pairs are the same.
3. Y_i is candidate.

We can prove the following theorem:

Theorem 6. *There is an upward embedding of flower f_c iff the corresponding condition graph has a directed cycle.*

The procedure we described above is repeated for every flower f_c. This procedure will find an upward embedding of the flower, if one exists. Notice that any upward embedding of the current flower will do since we can show that:

Lemma 7. *Consider current flower f_c, and let G be the original outerplanar graph. If G is upward planar, then for every upward embedding of f_c there is an upward embedding of G which respects the embedding of f_c.*

If no upward embedding can be found for the current flower, then the original outerplanar graph is not upward planar. Otherwise, DFS will continue with visiting the next face b (Fig. 5), and the same procedure will be applied. Notice that f_c with its petals are now a petal of current face b.

Theorem 8. *We can check in $O(n^2)$ time whether an odag has an upward embedding which is not topologically equivalent to the outerplanar embedding of the odag.*

Proof. Follows easily from the discussion above, and the fact that checking for a directed cycle in a condition graph has to be done at every current flower of the DFS process.

6 Conclusion and Open Problems

In this paper we solved the problem for testing upward planarity of odags. Our first algorithm checks whether an odag has an upward embedding which is equivalent to the outerplanar embedding of the odag. This is a linear time algorithm, and it is faster than any previously known technique. Our second algorithm checks whether an odag has an upward embedding which is not topologically equivalent to the outerplanar embedding. This algorithm is the first that deals with this problem, and solves it in polynomial time.

Our second algorithm is also the first work in this area that takes an approach of folding faces. It is open whether we can use a similar approach for other families of dags.

Since the general problem for checking upward planarity has been proved to be NP-complete, solving upward planarity efficiently for specific families of dags becomes interesting and important.

Acknowledgements

The author wishes to thank Seth Malitz for a variety of helpful discussions and especially for his comments on earlier drafts of the paper.

References

1. P. Bertolazzi and G. Di Battista, *On upward drawing testing of triconnected digraphs*, Proc. 7th ACM Symp. on Computational Geometry, pp. 272-280, 1991.
2. P. Bertolazzi, G. Di Battista, C. Mannino and R. Tamassia, *Optimal upward planarity testing of single-source digraphs*, In 1st Annual European Symp. on Algorithms, Lecture Notes in Comp. Sci., Springer-Verlag, 1993.
3. G. Di Battista, W.P. Liu, I. Rival, *Bipartite graphs, Upward drawings and Planarity*, Information Processing Letters, vol. 36, pp. 317-322, 1990.
4. G. Di Battista and R. Tamassia, *Algorithms for plane representations of acyclic digraphs*, Theoretical Computer Science, vol. 61, pp. 175-198, 1988.
5. H. de Fraysseix, J. Pach, R. Pollack, *How to draw a planar graph on a grid*, Combinatorica, vol. 10, pp. 41-51, 1990.
6. A. Garg and R. Tamassia, *On the Computational Complexity of Upward and Rectilinear Planarity Testing*, to appear in Proc. of Graph Drawing 1994.
7. J. Hopcroft and R. Tarjan, *Efficient planarity testing*, J. Assoc. Comput. Mach., vol. 21, pp. 549-568, 1974.
8. M.D. Hutton and A. Lubiw, *Upward planar drawing of single source acyclic digraphs*, Proc. 2nd ACM-SIAM Symp. on Discrete Algorithms, pp. 203-211, 1990.
9. D. Kelly, *On the dimension of partially ordered sets*, Discrete Math., vol. 63, pp. 197-216, 1987.
10. C. Platt, *Planar lattices and planar graphs*, J. Combin. Theory Ser. B, vol. 21, pp. 30-39, 1976.
11. W. Schnyder, *Embedding planar graphs on the grid*, Proc. 1st ACM-SIAM Symp. on Discrete Algorithms, pp. 138-147, 1990.