Parameterized Algorithms for Upward Planarity

Steven Chaplick
Maastricht University, The Netherlands

Emilio Di Giacomo
Università degli Studi di Perugia, Italy

Fabrizio Frati
Roma Tre University, Rome, Italy

Robert Ganian
Technische Universität Wien, Austria

Chrysanthi N. Raftopoulou
National Technical University of Athens, Greece

Kirill Simonov
Technische Universität Wien, Austria

Abstract

We obtain new parameterized algorithms for the classical problem of determining whether a directed acyclic graph admits an upward planar drawing. Our results include a new fixed-parameter algorithm parameterized by the number of sources, an XP-algorithm parameterized by treewidth, and a fixed-parameter algorithm parameterized by treedepth. All three algorithms are obtained using a novel framework for the problem that combines SPQR tree-decompositions with parameterized techniques. Our approach unifies and pushes beyond previous tractability results for the problem on series-parallel digraphs, single-source digraphs and outerplanar digraphs.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Human-centered computing → Graph drawings

Keywords and phrases Upward planarity, parameterized algorithms, SPQR trees, treewidth, treedepth

Digital Object Identifier 10.4230/LIPIcs.SoCG.2022.26

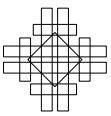
Related Version *Full Version*: `https://arxiv.org/abs/2203.05364`

Funding *Emilio Di Giacomo*: MIUR, grant 20174LF3T8, Dip. Ing. – UNIPG, grants RICBA19FM and RICBA20EDG.
Fabrizio Frati: MIUR, grant 20174LF3T8.
Robert Ganian: Austrian Science Fund (FWF) Project Y1329.
Chrysanthi N. Raftopoulou: NTUA research program ΠΕΒΕ 2020.
Kirill Simonov: Austrian Science Fund (FWF) Project P31336.

Acknowledgements The authors thank Fabrizio Montecchiani and Giuseppe Liotta for fruitful discussions on the topic of upward planarity. This research was initiated at Dagstuhl Seminar 21293: Parameterized Complexity in Graph Drawing [19].

1 Introduction

A digraph is called *upward planar* if it admits an upward planar drawing, that is, a planar drawing where all edges are oriented upward. The problem of upward planarity testing (Upward Planarity) and constructing an associated upward planar drawing arises, among others, in the context of visualization of hierarchical network structures; application domains include project management, visual languages and software engineering [2]. Upward planarity



is the most prominent notion of planarity that is inherently directed, and also has classical connections to the theory of ordered sets: the orders arising from the transitive closure of upward planar single-source digraphs have bounded dimension [30].

Since the introduction of the notion, Upward Planarity has become the focus of extensive theoretical research. The problem has been shown to be NP-complete more than 25 years ago [20, 21], but the first polynomial-time algorithms for restricted variants of Upward Planarity have been published even earlier [25, 26]. Among others, the problem is known to be polynomial-time tractable when G is provided with a planar embedding [3] (which also implies polynomial-time tractability for triconnected DAGs, since these admit a single planar embedding), or when restricted to the class of outerplanar DAGs [28], DAGs whose underlying graph is series-parallel [13], and most prominently single-source DAGs [2, 5, 26].

In spite of the number of results on Upward Planarity that analyze the classical complexity of the problem on specific subclasses of instances, the problem was up to now mostly unexplored from the more fine-grained perspective of parameterized complexity analysis [10, 15]. In particular, while it was known that Upward Planarity is fixed-parameter tractable when parameterized by the cyclomatic number of the input DAG (or, equivalently, the feedback edge number of the underlying undirected graph) [7], by the number of triconnected components and cut vertices [23], or the number of triconnected components plus the maximum diameter of a split component [13], the complexity of the problem under classical structural parameterizations has remained completely open.

Contribution. We develop a novel algorithmic framework for solving Upward Planarity which combines parameterized dynamic programming with the SPQR-tree decompositions of planar graphs [12, 22, 24]. In essence, our framework uses a characterization of the "shapes" of faces in an upward planar drawing that is inspired by earlier work on the notion of spirality [3, 13] and reduces Upward Planarity to the task of handling the "rigid" nodes in these decompositions. Informally, the task that needs to be handled there can be stated as follows: what are all the possible ways to combine the possible shapes of the children of a rigid node to obtain an upward planar drawing for the node itself? The framework is formalized in the form of a general "Interface Lemma" (Lemma 13) which can be complemented with numerous parameterizations as well as other algorithmic approaches.

In the remainder of this article, we use this framework to push the boundaries of tractability for Upward Planarity. Our first result in this direction is a fixed-parameter algorithm for Upward Planarity parameterized by the number of sources in the input graph. This result generalizes the polynomial-time tractability of the single-source case [2, 5] and answers an open question from a recent Dagstuhl seminar [19]. On a high level, we use the Interface Lemma to reduce the problem to a case where almost all children of a rigid node have a simple shape, and we show how this can be handled via a flow network approach.

Having established the tractability of instances with few sources, we turn towards understanding which structural properties of the underlying undirected graph can be used to solve Upward Planarity efficiently. In this context, apart from the fixed-parameter tractability of Upward Planarity parameterized by the feedback edge number [7], nothing was known about whether the more widespread "decompositional" parameters can be used to solve the problem. The parameters that will be of interest here are *treewidth* [29], the most prominent structural graph parameter, and *treedepth* [27], the arguably best known parameter that lies below treewidth in the parameter hierarchy (see, e.g., [1, Figure 1]).

To obtain new boundaries of tractability for Upward Planarity with respect to these two parameters, we first show that the problem posed by the Interface Lemma can be restated as a purely combinatorial problem on a suitable combinatorization of the embedding

of the graph represented by the rigid node, and – crucially – that a bound on the input graph's treewidth also implies a bound for the treewidth of this combinatorization. Once that is done, we design a non-trivial dynamic program that exploits this treewidth bound to handle the rigid nodes, which together with the Interface Lemma allows us to solve Upward Planarity. This yields an XP-algorithm for Upward Planarity parameterized by the treewidth of the underlying undirected graph – a result which unifies and generalizes the polynomial-time tractability of Upward Planarity on outerplanar as well as series-parallel graphs [13, 28]. Furthermore, a more detailed analysis of the dynamic program reveals that the same algorithm runs in fixed-parameter time when parameterized by treedepth.

Due to space limitations some proofs are omitted and can be found in [8].

2 Preliminaries

We refer to the usual sources for graph drawing and parameterized complexity terminology [10, 11, 14, 15]. We use $N_G(v)$ to denote the set of vertices adjacent to a vertex v in a graph G.

Upward planar drawings and embeddings. A *planar embedding* is an equivalence class of planar drawings of a graph, where two drawings are equivalent if the clockwise order of the edges incident to each vertex is the same and the outer faces are delimited by the same walk.

A vertex in a digraph is a *switch* if it is a source or a sink, and it is a *non-switch* otherwise. The *underlying graph* of a digraph is the undirected graph obtained from the digraph by ignoring the edge directions. A drawing of a digraph is *upward* if every edge is represented by a Jordan arc monotonically increasing from the source to the sink of the edge, and it is *upward planar* if it is both upward and planar. A digraph is *upward planar* if it admits an upward planar drawing; we use Upward Planarity to denote the problem of determining whether a digraph is upward planar; w.l.o.g., we assume that the input digraph is connected.

In an upward planar drawing Γ of a digraph G, an *angle* represents an incidence between a vertex v and a face f. The angle is either *flat* (if precisely one of the two edges incident to v and f is incoming at v), *large* (if v is a switch vertex and the angle has more than 180° in Γ), or *small* (otherwise) [3]; the latter two cases are jointly called *switch angles*. Then Γ defines an *angle assignment*, which assigns the value -1, 0, and 1 to each small, flat, and large angle, respectively, in every face of Γ. The angle assignment, together with the planar embedding of the underlying graph of G in Γ, constitutes an *upward planar embedding* of G.

The angle assignments that enhance a planar embedding into an upward planar embedding have been characterized by Didimo et al. [13], building on the work by Bertolazzi et al. [3]. Note that, once the planar embedding $\mathcal{E}$ of a digraph G is specified, then so are the angles of the faces of $\mathcal{E}$; in particular, whether an angle is flat or switch only depends on $\mathcal{E}$. Consider an angle assignment for $\mathcal{E}$. If v is a vertex of G, we denote by $n_i(v)$ the number of angles at v that are labeled i, with $i \in \{-1, 0, 1\}$. If f is a face of G, we denote by $n_i(f)$ the number of angles of f that are labeled i, with $i \in \{-1, 0, 1\}$. The cited characterization is as follows.

▶ **Theorem 1** ([3, 13]). *Let G be a digraph, $\mathcal{E}$ be a planar embedding of the underlying graph of G, and λ be an assignment of each angle of each face in $\mathcal{E}$ to a value in $\{-1, 0, 1\}$. Then $\mathcal{E}$ and λ define an upward planar embedding of G if and only if the following properties hold:*
UP0 *If α is a switch angle, then $\lambda(\alpha) \in \{-1, 1\}$, and if α is a flat angle, then $\lambda(\alpha) = 0$.*
UP1 *If v is a switch vertex of G, then $n_1(v) = 1$, $n_{-1}(v) = \deg(v) - 1$, $n_0(v) = 0$.*
UP2 *If v is a non-switch vertex of G, then $n_1(v) = 0$, $n_{-1}(v) = \deg(v) - 2$, $n_0(v) = 2$.*
UP3 *If f is a face of G, then $n_1(f) = n_{-1}(f) - 2$ if f is an internal face and $n_1(f) = n_{-1}(f) + 2$ if f is the outer face.*

Treewidth and Treedepth. Here we consider the treewidth and treedepth of the underlying graphs[1]. A *tree-decomposition* $\mathcal{T}$ of a graph $G = (V, E)$ is a pair (T, χ), where T is a tree (whose vertices we call *nodes*) rooted at a node r and χ is a function that assigns each node t a set $\chi(t) \subseteq V$ such that the following holds: for every $uv \in E$ there is a node t such that $u, v \in \chi(t)$, and for every vertex $v \in V$, the set of nodes t satisfying $v \in \chi(t)$ forms a nonempty subtree of T. The *width* of a tree-decomposition (T, χ) is the size of a largest set $\chi(t)$ minus 1, and the *treewidth* of the graph G, denoted $tw(G)$, is the minimum width of a tree-decomposition of G. The second structural parameter that we will be considering here is the *treedepth* of a graph G, denoted $td(G)$ [27]. A useful way of thinking about graphs of bounded treedepth is that they are (sparse) graphs with no long paths.

Expansion. In our algorithms, we will employ a linear-time preprocessing step called expansion to simplify the input digraphs so that every vertex has at most one incoming edge (in which case it is a *top* vertex) or at most one outgoing edge (in which case it is a *bottom* vertex) [2]. The expansion is obtained by replacing each non-switch vertex v with two new vertices v_1 and v_2, which inherit the incoming and outgoing edges of v, respectively, and the edge (v_1, v_2) (called the *special edge* of v_1 and v_2). It is known that expansion preserves upward planarity, and it is possible to observe that it preserves biconnectivity, does not create new sources, and only increases treewidth and treedepth by at most a factor of 2.

SPQR-tree decomposition. Let G be a biconnected undirected graph. A pair of vertices is a *separation pair* if its removal disconnects G. A *split pair* is either a separation pair or a pair of adjacent vertices. A *split component* of G with respect to a split pair $\{u, v\}$ is either an edge (u, v) or a maximal subgraph $G_{uv} \subset G$ such that $\{u, v\}$ is not a split pair of G_{uv}. A split pair $\{s', t'\}$ of G is *maximal* with respect to a split pair $\{s, t\}$ of G, if for every other split pair $\{s^*, t^*\}$ of G, there is a split component that includes the vertices s', t', s and t.

An *SPQR-tree* T of G with respect to an edge e^* is a rooted tree that describes a recursive decomposition of G induced by its split pairs [12]. Each node μ of T is associated with a split pair $\{u, v\}$ of G, where u and v are the *poles* of μ, with a subgraph G_μ of G, called the *pertinent graph* of μ, which consists of one or more split components of G with respect to $\{u, v\}$, and with a multigraph $\mathrm{sk}(\mu)$, called the *skeleton of* μ, which represents the arrangement of such split components in G_μ. The edges of $\mathrm{sk}(\mu)$ are called *virtual edges*. Each node μ of T whose pertinent graph is not a single edge has some children, each corresponding to a split components of G in G_μ. Each of these children is the root of a subtree of T. The nodes of T are of four types S, P, Q, and R. Q-nodes correspond to edges of G, while S-, P- and R-nodes correspond to so-called series, parallel and rigid compositions of the pertinent graphs of the children of the given node [12].

Note that each virtual edge e_i in the skeleton of a node μ of T *corresponds* to the pertinent graph G_{ν_i} of a child ν_i of μ. We say that G_{ν_i} is a *component* of G_μ. Figs. 1a and 1b show a planar graph and its SPQR-tree. To simplify our algorithms, we assume that every S-node of T has two children. If this is not the case, we can modify T to achieve this property (see Fig. 1c). An SPQR-tree T of an n-vertex planar graph has $\mathcal{O}(n)$ Q-, S-, P-, and R-nodes. Also, the total number of vertices of the skeletons for the nodes in T is $\mathcal{O}(n)$ [12].

When talking about an SPQR-tree T of a biconnected directed graph G, we mean an SPQR-tree of its underlying graph. Let μ be a node of T with poles u and v. A *uv-external upward planar embedding* of G_μ is an upward planar embedding of G_μ such that u and v

[1] Directed alternatives to treewidth exist, but are typically not well-suited for algorithmic applications [18].

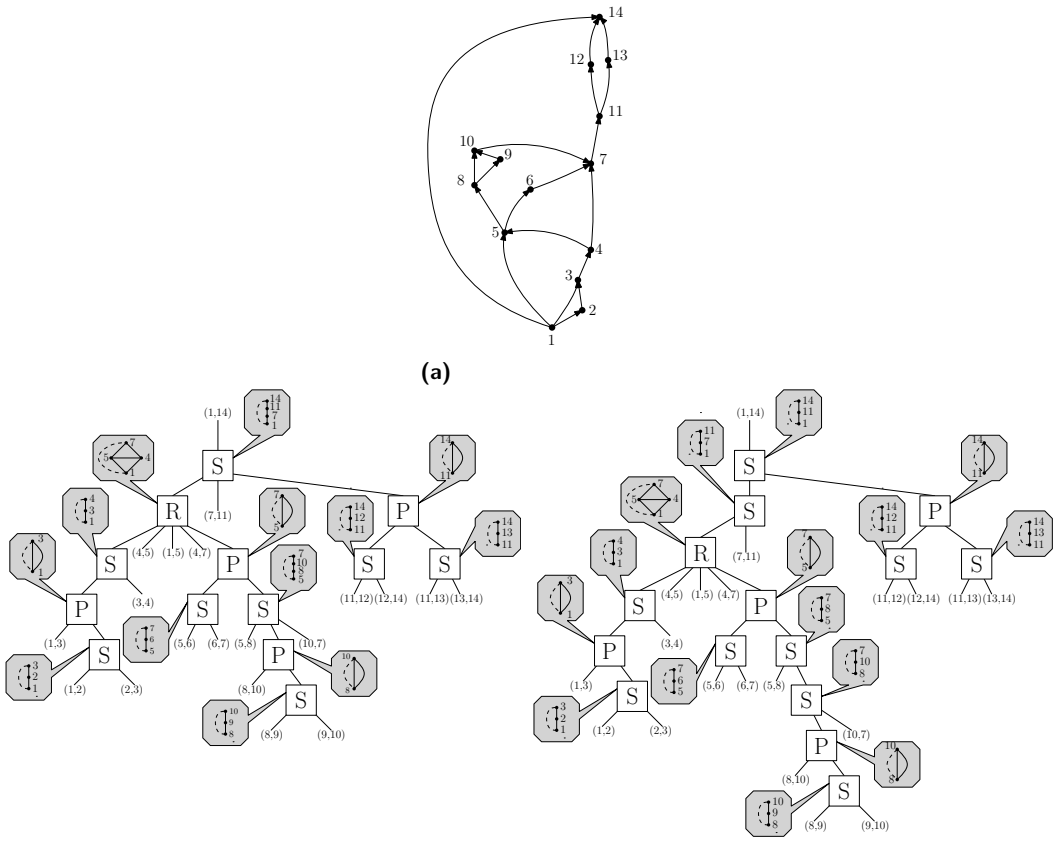


Figure 1 (a) A planar DAG G. (b) An SPQR-tree of G. For each node that is not a Q-node, the skeleton is depicted together with a dashed edge to represent the rest of the graph; for each Q-node, the corresponding edge is shown. (c) An SPQR-tree of G whose S-nodes have exactly two children.

are incident to the outer face. In our algorithms, when testing the upward planarity of a digraph G, the fact that its SPQR-tree T is rooted at an edge e^* of G corresponds to the requirement that e^* is incident to the outer face of the upward planar embedding $\mathcal{E}$ of G we are looking for. For each node μ of T, the restriction of $\mathcal{E}$ to the vertices and edges of the pertinent graph G_μ of μ is a uv-external upward planar embedding of G_μ.

3 The Shapes of Components

Let G be a biconnected DAG, let T be an SPQR-tree of G rooted at an edge e^*, let μ be a node of T with poles u and v, and let $\mathcal{E}_\mu$ be a uv-external upward planar embedding of G_μ. Let λ be the angle assignment defined by $\mathcal{E}_\mu$. The poles u and v identify two paths on the boundary of the outer face f_0 of $\mathcal{E}_\mu$: the *left outer path* $P_l = \langle v_0 = u, v_1, \ldots, v_k = v \rangle$ is the path that leaves f_0 on the left when walking from u to v; the *right outer path* $P_r = \langle w_0 = u, w_1, \ldots, w_h = v \rangle$ of $\mathcal{E}_\mu$ is the path that leaves f_0 on the right when walking from u to v; see Fig. 2. For $i = 0, 1, \ldots, k$, let α_i denote the angle at v_i inside f_0 and, for $i = 0, 1, \ldots, h$, let β_i denote the angle at w_i inside f_0. The *left-turn-number* $\tau_l(\mathcal{E}_\mu, u, v)$ of $\mathcal{E}_\mu$ is defined as $\sum_{i=1}^{k-1} \lambda(\alpha_i)$, while the *right-turn-number* $\tau_r(\mathcal{E}_\mu, u, v)$ of $\mathcal{E}_\mu$ is $\sum_{i=1}^{h-1} \lambda(\beta_i)$. Note that $\alpha_0 = \beta_0$ and $\alpha_k = \beta_h$ are the angles at u and v inside f_0, respectively. The values $\lambda(\alpha_0)$ and $\lambda(\alpha_k)$ are also denoted by

$\lambda(\mathcal{E}_\mu, u)$ and $\lambda(\mathcal{E}_\mu, v)$, respectively. Finally, given a vertex $w \in \{u, v\}$, let $\rho_l(\mathcal{E}_\mu, w)$ denote the orientation of the edge e_l of P_l incident to w, that is, $\rho_l(\mathcal{E}_\mu, w) = in$ if e_l is an incoming edge for w, $\rho_l(\mathcal{E}_\mu, w) = out$ otherwise. Analogously, let $\rho_r(\mathcal{E}_\mu, w)$ denote the orientation of the edge e_r of P_r incident to w. The *shape description of* $\mathcal{E}_\mu$ is the tuple $\langle \tau_l(\mathcal{E}_\mu, u, v), \tau_r(\mathcal{E}_\mu, u, v), \lambda(\mathcal{E}_\mu, u), \lambda(\mathcal{E}_\mu, v), \rho_l(\mathcal{E}_\mu, u), \rho_r(\mathcal{E}_\mu, u), \rho_l(\mathcal{E}_\mu, v), \rho_r(\mathcal{E}_\mu, v)\rangle$; see Fig. 2.

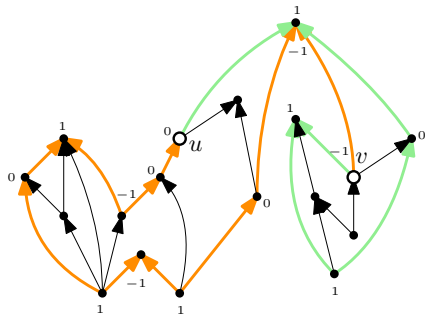


Figure 2 An upward planar embedding of a split component G_μ with poles u and v and shape description $\langle 3, 0, 0, -1, out, in, out, out\rangle$. The left (right) outer path is shown in green (orange).

There are some dependencies between the values of a shape description. For example, $\rho_l(\mathcal{E}_\mu, u) \neq \rho_r(\mathcal{E}_\mu, u)$ if $\lambda(\mathcal{E}_\mu, u) = 0$. As a further example, we have the following observation, which comes from Property **UP3** of Theorem 1 and uses the notation of this theorem.

▶ **Observation 2.** *We have* $\tau_l(\mathcal{E}_\mu, u, v) + \tau_r(\mathcal{E}_\mu, u, v) + \lambda(\mathcal{E}_\mu, u) + \lambda(\mathcal{E}_\mu, v) = 2$.

Recall that if u is a top or bottom vertex of G, then it has at most one incoming edge or at most one outgoing edge, respectively, which is called the *special edge* of u. If G_μ contains this edge, then G_μ is a *special* component for u, otherwise we say that G_μ is a *normal* component for u. Note that, if u is a source or a sink of G, then it has no special component.

▶ **Lemma 3.** *We have* $\tau_r(\mathcal{E}_\mu, u, v) = -\tau_l(\mathcal{E}_\mu, u, v) + h$, *with* $h \in \{0, 1, 2, 3, 4\}$.

4 General Algorithm

Let G be an n-vertex biconnected expanded DAG whose underlying graph is planar and let T be an SPQR-tree of G. Let $\tau_{\min}$ and $\tau_{\max}$ be two integers with $\tau_{\min} \leq \tau_{\max}$ and let $\tau = \tau_{\max} - \tau_{\min} + 1$. We present a general algorithm to compute all possible shape descriptions of G with respect to T, and such that the left- and right-turn numbers of all shape descriptions for all pertinent graphs of T are in the range $[\tau_{\min}, \tau_{\max}]$. We visit the nodes of T bottom-up and we compute for each node μ its *feasible set* $\mathcal{F}_\mu$, i.e., the set of all realizable shape descriptions of its pertinent graph G_μ. If $\mathcal{F}_\mu = \emptyset$, the process stops and G is not upward planar (under the above restrictions), otherwise we continue the traversal of T.

Storing feasible sets. For each node μ of T we associate a matrix $M(\mu)$ of size $(\tau_{\max} - \tau_{\min} + 1) \times 5$ where the element $M(\mu)[i, j]$ of the matrix contains all shape descriptions of G_μ with left turn-number $\tau_l = \tau_{\min} + i$ and right-turn-number $\tau_r = -\tau_l + j$. Note that by Lemma 3, τ_r can only take values in $[-\tau_l, -\tau_l + 4]$.

▶ **Lemma 4.** *There are at most* 18 *shape descriptions with given left- or right-turn-number.*

We describe how to compute the feasible set $\mathcal{F}_\mu$ of a node μ of T depending on its type.

Q-node. The pertinent graph G_μ is either the edge (u, v) or (v, u). Hence, the feasible set consists of the tuple $\langle 0, 0, 1, 1, out, out, in, in\rangle$ or $\langle 0, 0, 1, 1, in, in, out, out\rangle$, respectively.

▶ **Lemma 5.** *Let μ be a Q-node of T. The feasible set $\mathcal{F}_\mu$ can be computed in $\mathcal{O}(1)$ time.*

S-node. Let ν_1 and ν_2 be the children of μ, with poles u, w and w, v, respectively. Let $\langle \tau_l^1, \tau_r^1, \lambda_u^1, \lambda_w^1, \rho_{l,u}^1, \rho_{r,u}^1, \rho_{l,w}^1, \rho_{r,w}^1\rangle$ be a tuple in $\mathcal{F}_{\nu_1}$ and let $\langle \tau_l^2, \tau_r^2, \lambda_w^2, \lambda_v^2, \rho_{l,w}^2, \rho_{r,w}^2, \rho_{l,v}^2, \rho_{r,v}^2\rangle$ be a tuple in $\mathcal{F}_{\nu_2}$. Let α_l (resp. α_r) be the angle at w created by the two left (resp. right) outer paths of G_{ν_1} and G_{ν_2} (see Fig. 3). We assign the labels λ_l and λ_r to α_l and α_r respectively as follows: $\lambda_l = 0$ if $\rho_{l,w}^1 \neq \rho_{l,w}^2$ otherwise $\lambda_l \in \{-1, 1\}$, and $\lambda_r = 0$ if $\rho_{r,w}^1 \neq \rho_{r,w}^2$ otherwise $\lambda_r \in \{-1, 1\}$. Note that, by **UP1** it must be $\lambda_l + \lambda_r < 2$. For all possible values of λ_l and λ_r satisfying the previous constraints, we construct a candidate tuple $\langle \tau_l, \tau_r, \lambda_u, \lambda_v, \rho_{l,u}, \rho_{r,u}, \rho_{l,v}, \rho_{r,v}\rangle$ with: (i) $\tau_l = \tau_l^1 + \tau_l^2 + \lambda_l$, (ii) $\tau_r = \tau_r^1 + \tau_r^2 + \lambda_r$, (iii) $\lambda_u = \lambda_u^1$, (iv) $\lambda_v = \lambda_v^2$, (v) $\rho_{l,u} = \rho_{l,u}^1$, (vi) $\rho_{r,u} = \rho_{r,u}^1$, (vii) $\rho_{l,v} = \rho_{l,v}^2$, (viii) $\rho_{r,v} = \rho_{r,v}^2$. We accept the candidate tuple if and only if it satisfies Observation 2.

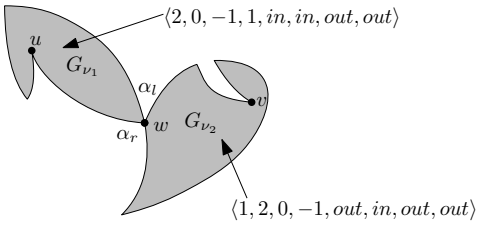


Figure 3 Series composition. The resulting shape description is $\langle 2, 2, -1, -1, in, in, out, out\rangle$.

▶ **Lemma 6.** *Let μ be an S-node of T with children ν_1 and ν_2. The feasible set $\mathcal{F}_\mu$ can be computed in $\mathcal{O}(\tau + |\mathcal{F}_{\nu_1}| \cdot |\mathcal{F}_{\nu_2}|)$ time.*

P-node. Let μ be a P-node with poles u and v and k children $\nu_1, \nu_2 \ldots, \nu_k$. Let N' be a subset of the children of μ and let G'_μ be the subgraph of G_μ consisting of components $G_{\nu'}$ for $\nu' \in N'$. Consider a uv-external upward planar embedding $\mathcal{E}'_\mu$ of G'_μ. Denote by S' the sequence of shape descriptions of the components of G'_μ in the clockwise order in which they appear around u starting from the outer face. The sequence S' is the *shape sequence* of G'_μ with respect to $\mathcal{E}'_\mu$. To describe S' we write: a^* (resp. a^+) to denote a subsequence of S' consisting of 0 (resp. 1) or more elements equal to a. We say that a shape description s' of G'_μ *corresponds* to S' if there exists an upward planar embedding of G'_μ with shape description s' and whose shape sequence is S'. Let S be a sequence of shape descriptions; the *reduced sequence* of S is obtained from S by replacing each maximal subsequence a^+ of S with the single element a. The *size* of S is the number of elements in its reduced sequence.

▶ **Lemma 7.** *Let S' be a sequence of shape descriptions from the feasible sets of every $G_{\nu'}$, with $\nu' \in N'$. We can decide whether S' is a shape sequence of G'_μ and compute the corresponding shape descriptions of G'_μ in $\mathcal{O}(r^3)$ time, where r is the size of S'. Furthermore there are $\mathcal{O}(r^2)$ computed shape descriptions of G'_μ.*

Let ν be a child of G_μ with $\nu \notin N'$, let s be a shape description of G_ν, and let G''_μ be the union of G_ν and G'_μ. We say that S' *can be extended* with s to a shape sequence S'' of G''_μ if S'' is a shape sequence of G''_μ, s belongs to S'', and removing s from S'' we obtain S'.

▶ **Lemma 8.** *Let S' be a shape sequence of G'_μ. Given a shape description s of G_ν, we can decide whether S' can be extended with s to a shape sequence S'' of G''_μ and compute the corresponding shape descriptions of G''_μ in $\mathcal{O}(r^4)$ time, where r is the size of S'.*

Suppose that G_μ is upward planar and consider a uv-external upward planar embedding $\mathcal{E}_\mu$ of G_μ. We remove the special components of u and v and the normal components G_ν whose shape description labels the angle at u or v with -1. There are at most two such components, as each one labels an internal angle at a pole with 1. Let G'_μ be the subgraph of G_μ obtained after this removal; G'_μ is the *thin subgraph* of G_μ with respect to $\mathcal{E}_\mu$. In the next lemma, if $w \in \{u, v\}$ is a top vertex then $\rho_w = out$, otherwise $\rho_w = in$.

▶ **Lemma 9.** *Let μ be a P-node such that G_μ is upward planar and let $\mathcal{E}_\mu$ be a uv-external upward planar embedding of G_μ such that the left-turn-number of G'_μ is c. Then the shape sequence of G'_μ with respect to $\mathcal{E}_\mu$ is $[s_1{}^+, s_2{}^*, s_3{}^*]$, with $s_1 = \langle c, -c, 1, 1, \rho_u, \rho_u, \rho_v, \rho_v\rangle$, $s_2 = \langle c-2, -c+2, 1, 1, \rho_u, \rho_u, \rho_v, \rho_v\rangle$, $s_3 = \langle c-4, -c+4, 1, 1, \rho_u, \rho_u, \rho_v, \rho_v\rangle$.*

Based on Lemma 9, our algorithm computes in three steps the shape descriptions of G_μ that match some fixed left-turn-number c_l and right-turn-number c_r. Let c'_l be equal to c_l or $c_l - 1$, depending on whether exactly one of u and v is a bottom vertex or not. For the first step, we consider all sequences $S' = [s_1{}^*, s_2{}^*, s_3{}^*]$ where $s_i = \langle c'_l - 2(i-1), -c'_l + 2(i-1), 1, 1, \rho_u, \rho_u, \rho_v, \rho_v\rangle$, for $i = 1, 2, 3$. For each of them we identify a maximal subgraph G'_μ of G_μ such that S' is a shape sequence of G'_μ. For each child ν_i of μ (with $i = 1, 2, \ldots, k$), we check whether the feasible set $\mathcal{F}_{\nu_i}$ contains shape descriptions of S' in the order that they appear in S'; if so, we choose it for G_{ν_i}. This greedy process does not necessarily produce the desired sequence S'. By reassigning at most two components of G'_μ either we get S' or no subgraph G'_μ has S' as its shape sequence.

For the second step, we focus on the children of μ that, when considering a shape sequence S', have not been assigned a shape description so far. There are at most two such children, say ν and ν', otherwise G_μ does not admit an upward planar embedding whose thin subgraph has S' as its shape sequence. Let s_ν (resp. $s_{\nu'}$) be a shape description in $\mathcal{F}_\nu$ (resp. $\mathcal{F}_{\nu'}$). Using Lemma 8 we compute all possible extensions of S' with s_ν and $s_{\nu'}$ to shape sequences of G_μ (in $\mathcal{O}(1)$ time since the size r of S' is at most 3). For every computed shape sequence S of G_μ we check whether it matches c_l and c_r. If so, we add to $\mathcal{F}_\mu$ all shape descriptions of G_μ that correspond to S (in $\mathcal{O}(1)$ time since the size r of S is at most 5). Otherwise, we proceed to the third step with S.

To complete the procedure, we perform a case analysis to handle situations where one or both of c_l and c_r are not matched. Intuitively, our goal is to find a component of the thin subgraph G'_μ, remove its current shape description from S and use another one from its feasible set at the beginning or at the end of the sequence in order to match c_l or c_r. If none of the components of G'_μ can be used for this purpose, we conclude that the pair c_l and c_r cannot be realized. Otherwise, using Lemma 7 (where the size r is at most 5), we compute all corresponding shape descriptions of G_μ and add them to $\mathcal{F}_\mu$.

▶ **Lemma 10.** *Let μ be an P-node of T with k children. The feasible set $\mathcal{F}_\mu$ can be computed in $\mathcal{O}(\tau \cdot k)$ time.*

R-node. The R-nodes will be handled differently in Sections 6 and 7 depending on the parameter we use. To complete the description of our framework we introduce the notion of an R-node subprocedure. Formally, an *R-node subprocedure* is an algorithm which takes as

input an R-node μ of T and a mapping $\mathcal{S}_\mu$ which assigns each child of μ to its feasible set, and computes the feasible set $\mathcal{F}_\mu$ in at most $\alpha(G_\mu, \mathcal{S}_\mu)$ time. For a DAG G with SPQR-tree T, $\alpha(G) = \sum_{\text{R-node } \mu} \alpha(G_\mu, \mathcal{S}_\mu)$ is the *total time complexity* of the R-node subprocedure for G.

Root node. The root node r corresponds to an edge $e = (u, v)$ of G that lies on its outer face and has only one child μ with poles u and v. Since $G_\mu = G \setminus e$, node r can be treated as a P-node with poles u and v and two children; one of them is μ and the other one is a Q-node for the edge (u, v). By Lemma 10, we can compute the feasible set of r in $\mathcal{O}(\tau)$ time.

▶ **Lemma 11.** *The feasible set $\mathcal{F}_r$ of the root node r of T can be computed in $\mathcal{O}(\tau)$ time.*

Combining Lemmata 5, 6, 10 and 11, we obtain the following lemma.

▶ **Lemma 12.** *Let G be a biconnected DAG with n vertices and let T be an SPQR-tree of G rooted at a Q-node corresponding to an edge $e = (u, v)$. Let $\tau_{\min}$ and $\tau_{\max}$ be two given integer values, and let $\tau = \tau_{\max} - \tau_{\min} + 1$. Given an R-node subprocedure with total time complexity $\alpha(G)$, it is possible to compute in time $\mathcal{O}(\alpha(G) + \tau^2 \cdot n)$ the shape descriptions of every upward planar embedding with e on the outer face, such that the left- and right-turn-numbers of the pertinent graph of every node of T are in the range $[\tau_{\min}, \tau_{\max}]$.*

5 Extension to the Single-Connected Case

In this section, we establish the Interface Lemma, which reduces the task of solving Upward Planarity to the one of obtaining an R-node subprocedure, for *all* graphs including single-connected ones. To formalize the lemma, we call a digraph G $[\tau_{\min}, \tau_{\max}]$-*turn-bounded* if every upward planar embedding of G has the following property: the pertinent graphs of any SPQR-tree of each biconnected component in G have left- and right-turn numbers in the range $[\tau_{\min}, \tau_{\max}]$.

▶ **Lemma 13** (Interface Lemma). *Let G be an n-vertex digraph, and $\tau_{\min}$, $\tau_{\max}$ be integers such that G is $[\tau_{\min}, \tau_{\max}]$-turn-bounded. Given an R-node subprocedure with total time complexity $\alpha(G)$, it is possible to determine whether G admits an upward planar embedding in time $\mathcal{O}(n(\alpha(G) + \tau^2 \cdot n))$ where $\tau = \tau_{\max} - \tau_{\min} + 1$.*

Note that, for a single-connected graph G, we define the total time complexity $\alpha(G)$ of an R-node subprocedure to be the sum of $\alpha(B)$ over all biconnected components B of G.

To give an intuition of the proof, consider a fixed rooting of the block-cut tree of G. The core of our algorithm is a procedure that, given suitable embeddings of leaf components containing the same cut-vertex, attaches these embeddings to an arbitrary upward planar embedding of the rest of the graph. This allows us to process the block-cut tree upwards: we iteratively verify that there exist desired embeddings for a group of leaf blocks via the biconnected algorithm (Lemma 12), and reduce to a smaller tree by removing these blocks.

6 An Algorithm Parameterized by the Number of Sources

Let G be an acyclic digraph with n vertices and σ sources, whose underlying graph is planar. In order to obtain an algorithm for Upward Planarity parameterized by σ, in view of Lemma 13, we devise an R-node subprocedure whose runtime depends on σ and, polynomially, on n. We hence assume that G is biconnected and that has been expanded. Let e^* be any edge of G; we compute an SPQR-tree T of G rooted at the Q-node representing e^* in $\mathcal{O}(n)$ time [12, 22]. A key ingredient of our algorithm is the following.

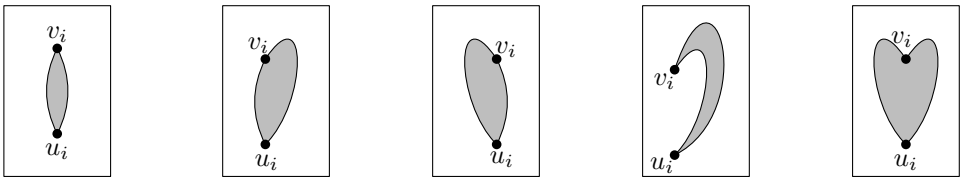


Figure 4 Shape descriptions of boring components.

▶ **Lemma 14.** *Let μ be a node of T, let u and v be the poles of μ, let σ_μ be the number of sources of G_μ different from its poles, and let $\mathcal{E}_\mu$ be any uv-external upward planar embedding of G_μ. The left- and right-turn-numbers of $\mathcal{E}_\mu$ are in the interval $[-2\sigma_\mu - 1, 2\sigma_\mu + 1]$. Furthermore, the size of the feasible set $\mathcal{F}_\mu$ of μ is at most $72\sigma_\mu + 54$.*

Let μ be an R-node of T with children $\nu_1, \ldots, \nu_k$. Let u and v be the poles of μ, σ_μ be the number of sources of G_μ different from its poles; for $i = 1, \ldots, k$, let u_i and v_i be the poles of ν_i and e_i be the virtual edge representing ν_i in the skeleton $\mathrm{sk}(\mu)$ of μ. We give an algorithm that computes $\mathcal{F}_\mu$ from the feasible sets $\mathcal{F}_{\nu_1}, \ldots, \mathcal{F}_{\nu_k}$ in $\mathcal{O}(\sigma 1.45^\sigma \cdot k \log^3 k)$ time.

We introduce two classifications of the components of G_μ. A component G_{ν_i} is *interesting* if it contains sources other than its poles, and *boring* otherwise. Because G has σ sources, at most σ components among $G_{\nu_1}, \ldots, G_{\nu_k}$ are interesting, while any number of components can be boring. Second, a component G_{ν_i} is *extreme* if e_i is incident to a pole of μ and is incident to the face containing u and v of any planar embedding of $\mathrm{sk}(\mu)$, and *non-extreme* otherwise. Note that there are four extreme components among $G_{\nu_1}, \ldots, G_{\nu_k}$, because there are exactly two virtual edges incident to each of u and v in the considered face. We can order $G_{\nu_1}, \ldots, G_{\nu_k}$ in $O(k \log k)$ time so that all the extreme or interesting components come first.

Despite their name, boring components play an important role in our algorithm. A key feature is that a u_iv_i-external upward planar embedding of a boring component G_{ν_i} can only have one of $O(1)$ shape descriptions: the **sausage** $\langle 0, 0, 1, 1, \mathrm{out}, \mathrm{out}, \mathrm{in}, \mathrm{in}\rangle$, see Fig. 4a; the **inverted-sausage** $\langle 0, 0, 1, 1, \mathrm{in}, \mathrm{in}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4a with u_i and v_i inverted; the **right-wing** $\langle 0, 1, 1, 0, \mathrm{out}, \mathrm{out}, \mathrm{in}, \mathrm{out}\rangle$, see Fig. 4b; the **inverted-right-wing** $\langle 1, 0, 0, 1, \mathrm{out}, \mathrm{in}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4b with u_i and v_i inverted; the **left-wing** $\langle 1, 0, 1, 0, \mathrm{out}, \mathrm{out}, \mathrm{out}, \mathrm{in}\rangle$, see Fig. 4c; the **inverted-left-wing** $\langle 0, 1, 0, 1, \mathrm{out}, \mathrm{in}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4c with u_i and v_i inverted; the **hat** $\langle -1, 1, 1, 1, \mathrm{out}, \mathrm{out}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4d; the **inverted-hat** $\langle 1, -1, 1, 1, \mathrm{out}, \mathrm{out}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4d with u_i and v_i inverted; the **heart** $\langle 1, 1, 1, -1, \mathrm{out}, \mathrm{out}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4e; and the **inverted-heart** $\langle 1, 1, -1, 1, \mathrm{out}, \mathrm{out}, \mathrm{out}, \mathrm{out}\rangle$, see Fig. 4e with u_i and v_i inverted. Furthermore, we can prove that not all such shape descriptions can occur simultaneously in the feasible set of a node ν_i and that some shape descriptions are "better" than others. This allows us to assume that the feasible set of a node ν_i contains: only the sausage, or only the inverted-sausage, or only the left-wing and the right-wing, or only the inverted-left-wing and the inverted-right-wing, or only the hat and the inverted-hat, or only the heart, or only the inverted-heart, or only the heart and the inverted-heart.

We test independently whether each shape description $s = \langle \tau_l, \tau_r, \lambda_u, \lambda_v, \rho_{l,u}, \rho_{r,u}, \rho_{l,v}, \rho_{r,v}\rangle$, where $\tau_l \in [-2\sigma_\mu - 1, 2\sigma_\mu + 1]$, $\tau_r \in [-\tau_l, -\tau_l + 4]$, $\lambda_u \in \{-1, 0, 1\}$, $\lambda_v \in \{-1, 0, 1\}$, $\rho_{l,u} \in \{\mathrm{in}, \mathrm{out}\}$, $\rho_{r,u} \in \{\mathrm{in}, \mathrm{out}\}$, $\rho_{l,v} \in \{\mathrm{in}, \mathrm{out}\}$, and $\rho_{r,v} \in \{\mathrm{in}, \mathrm{out}\}$ belongs to $\mathcal{F}_\mu$ or not. Note that $\tau_l \in [-2\sigma_\mu - 1, 2\sigma_\mu + 1]$ and $\tau_r \in [-\tau_l, -\tau_l + 4]$ can be assumed without loss of generality by Lemmata 14 and 3, respectively, thus the number of shape descriptions to be tested is in $\mathcal{O}(\sigma_\mu)$. We select shape descriptions $s_1 \in \mathcal{F}_{\nu_1}, \ldots, s_h \in \mathcal{F}_{\nu_h}$ for the extreme or interesting components $G_{\nu_1}, \ldots, G_{\nu_h}$ of G_μ. Clearly, the number ℓ of ways this selection can

be done is $\ell = \prod_{i=1}^{h} |\mathcal{F}_{\nu_i}|$; by exploiting the bound on $|\mathcal{F}_{\nu_i}|$ given by Lemma 14, we can prove that $\ell \in \mathcal{O}(1.45^{\sigma})$. We also fix $\mathcal{S}_\mu$ to be a planar embedding of the skeleton $\mathrm{sk}(\mu)$ of μ in which u and v are incident to the outer face. Since μ is an R-node, there are two such planar embeddings, which are flips of each other. The goal now becomes the one of testing whether G_μ admits a uv-external upward planar embedding $\mathcal{E}_\mu$ such that: (P1) the shape description of $\mathcal{E}_\mu$ is s; (P2) for $i = 1, \ldots, h$, the $u_i v_i$-external upward planar embedding $\mathcal{E}_{\nu_i}$ of G_{ν_i} in $\mathcal{E}_\mu$ has shape description s_i; and (P3) the planar embedding of $\mathrm{sk}(\mu)$ induced by $\mathcal{E}_\mu$ is $\mathcal{S}_\mu$. Then we have that s belongs to $\mathcal{F}_\mu$ if and only if this test is successful for at least one selection of the shape descriptions $s_1, \ldots, s_h$ and of the planar embedding $\mathcal{S}_\mu$.

We now borrow ideas from an algorithm by Bertolazzi et al. [3] for testing the upward planarity of a digraph D with a prescribed planar embedding $\mathcal{E}$. The algorithm in [3] constructs a bipartite flow network $\mathcal{N}(S, T, A)$, where each source $s_w \in S$ corresponds to a switch vertex w of D, each sink $t_f \in T$ corresponds to a face f of $\mathcal{E}$, and A has an arc from s_w to t_f if w is incident to f. A unit of flow passing from s_w to t_f corresponds to a large angle at w in f. Each source supplies 1 unit of flow, each arc has capacity 1, and each sink t_f demands $n_f/2 - 1$ units of flow if f is an internal face of $\mathcal{E}$ and $n_f/2 + 1$ if f is the outer face of $\mathcal{E}$, where n_f is the number of switch angles incident to f. Then D has an upward planar embedding which respects $\mathcal{E}$ if and only if $\mathcal{N}$ has a flow in which each sink is supplied with a number of units of flow equal to its demand.

After some preliminary checks, which ensure that the values $s, s_1, \ldots, s_h, \mathcal{S}_\mu$ are "coherent" with each other, we also construct a flow network $\mathcal{N}(S, T, A)$. Note that the skeleton $\mathrm{sk}(\mu)$ of our R-node μ has a prescribed planar embedding $\mathcal{S}_\mu$. However, the edges of $\mathrm{sk}(\mu)$ are not actual edges, but rather virtual edges that correspond to components of G_μ. These components introduce new sources, sinks, and arcs in $\mathcal{N}$, and contribute to the demands of their incident faces. As we have already fixed the shape description s_i of each extreme or interesting component G_{ν_i}, we know the excess of large angles with respect to small angles "on the sides" of G_{ν_i}, as these are the first two values of s_i. These values introduce sources (if they are positive) and contribute to the demands of the faces of $\mathcal{S}_\mu$ incident to e_i. Handling non-extreme boring components is more challenging. Each boring component has at most two shape descriptions in its feasible set, however the number of such components is not, in general, bounded by a function of σ only, hence we cannot try all possible combinations for their shape descriptions. Rather, we plug the freedom of choosing a shape description for each non-extreme boring component directly into the flow network. For example, a component G_{ν_i} such that $\mathcal{F}_{\nu_i}$ contains the hat and the inverted-hat is modeled by a source with two incident arcs to the faces of $\mathcal{S}_\mu$ incident to e_i, reflecting the fact that each of the two shape descriptions provides a large angle in a different face incident to e_i. As another example, a component G_{ν_i} such that $\mathcal{F}_{\nu_i}$ contains the left-wing and the right-wing also provides a large angle in a different face incident to e_i depending on the choice of the shape description, however in this case the choice might also affect whether a pole of the component creates a switch angle in a face of $\mathcal{S}_\mu$ or not, which affects the demand of the face. This is solved either by "ignoring" the component, or by transfering its effect to an adjacent non-switch vertex.

Figure 5 shows an example of the construction of $\mathcal{N}$. We have that $\mathcal{N}$ has $\mathcal{O}(k)$ nodes and arcs. We test whether every sink has a non-negative demand and whether $\mathcal{N}$ admits a flow in which every sink receives an amount of flow equal to its demand. The latter can be done in $\mathcal{O}(k \log^3 k)$ time by means of an algorithm by Borradaile et al. [4]. We conclude that G_μ admits a uv-external upward planar embedding satisfying Properties P1–P3 if and only if the tests are successful. This leads to the following.

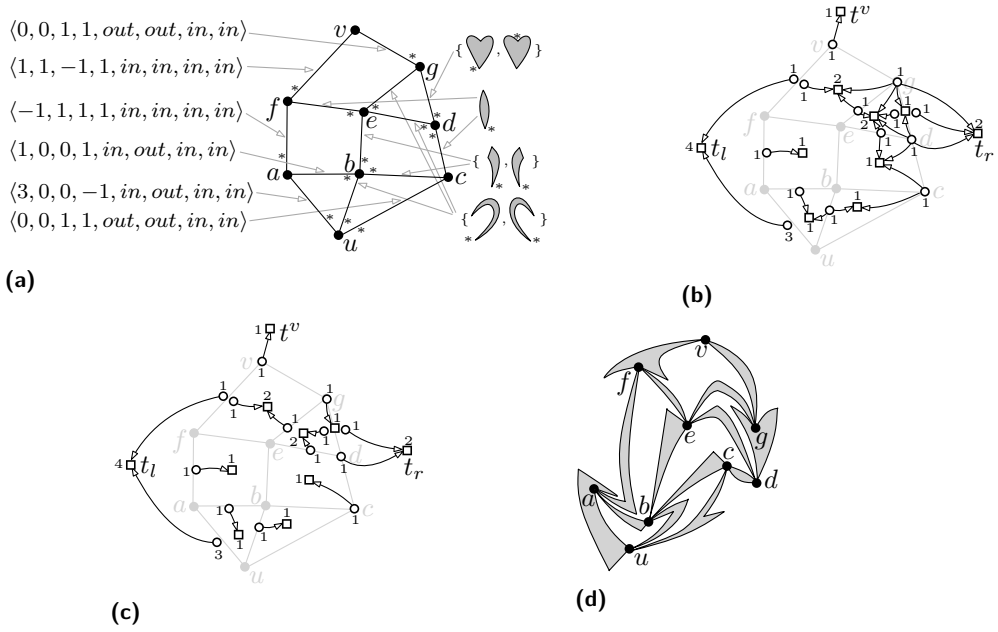


Figure 5 The construction of a flow network $\mathcal{N}$ that allows us to determine whether a shape description $s = \langle 1, 0, -1, 0, \text{in}, \text{out}, \text{in}, \text{in}\rangle$ belongs to $\mathcal{F}_\mu$. (a) shows the input: a shape description s_i for each extreme or interesting component G_{ν_i} of G_μ and the feasible set $\mathcal{F}_{\nu_i}$ for each non-extreme boring component G_{ν_i} of G_μ. (b) shows $\mathcal{N}$; arc capacities are not shown (each of them is equal to the supply of the source of the arc). (c) shows a flow for $\mathcal{N}$ in which every sink receives an amount of flow equal to its demand; each shown arc is traversed by a flow equal to its capacity. (d) shows a uv-external upward planar embedding of G_μ with shape description s corresponding to the flow.

▶ **Lemma 15.** *The feasible set $\mathcal{F}_\mu$ of an R-node μ of T can be computed in $\mathcal{O}(\sigma 1.45^\sigma \cdot k \log^3 k)$ time, where k is the number of children of μ in T and σ is the number of sources of G.*

Lemmata 13 and 15 imply the following main result.

▶ **Theorem 16.** Upward Planarity *can be solved in $\mathcal{O}(\sigma 1.45^\sigma \cdot n^2 \log^3 n)$ time for a digraph with n vertices and σ sources.*

7 An Algorithm Parameterized by Treewidth

The aim of this section is to provide an R-node subprocedure which yields parameterized algorithms for Upward Planarity when parameterized by treewidth and treedepth. The idea behind this is to obtain a combinatorization of the task that is asked in the subprocedure. This will be done by extending the skeleton of the R-node with additional information, notably via a so-called *embedding graph*[2]. The R-node subprocedure is then obtained by performing dynamic programming over the embedding graph. However, to obtain the desired runtime, we will first have to show that the embedding graph has bounded treewidth.

[2] Note that this notion differs from the embedding graphs used in recent drawing extension problems [16, 17]; unlike in those problems, here it seems impossible to use Courcelle's Theorem [9].

A Combinatorial Representation of the Skeleton. Let G be a connected graph with a planar embedding $\mathcal{G}$, and let F be the set of faces of $\mathcal{G}$. Let G^- be the graph obtained from G by subdividing each edge e once, creating the vertex v_e. We define the *embedding graph* $\widetilde{G}$ of G as the graph obtained from G^- by adding a vertex f for each face in F, and connecting f to each vertex in G^- incident to f. Observe that $\widetilde{G}$ is tripartite, and we call the three sets of vertices that occur in the definition of $V(\widetilde{G})$ the *true vertices*, *face-vertices* and *edge-vertices* of $\widetilde{G}$, respectively. An illustration is shown in Fig. 6.

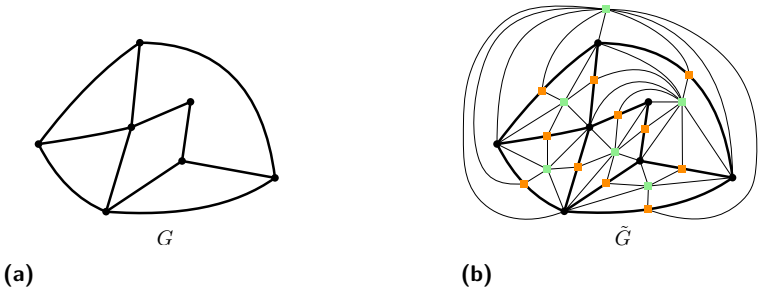


Figure 6 (a) A planar graph G. (b) The embedding graph $\tilde{G}$ of G. True-, face-, and edge-vertices are shown in black, green, and orange, respectively.

Our aim in this section is to show that $tw(\widetilde{G})$ is linearly bounded by $tw(G)$. To do so, we identify the faces that are, in some sense, "relevant" for a bag in a tree decomposition of G^-, and prove that (1) the number of such relevant faces is linearly bounded by the width of that decomposition and (2) adding these faces to the decomposition of G^- results in a tree-decomposition of $\widetilde{G}$. We can then prove:

▶ **Theorem 17.** *Let G be a graph with a planar embedding of treewidth k where $k \geq 1$. Then the embedding graph $\widetilde{G}$ has treewidth at most $11k - 4 \in \mathcal{O}(k)$.*

Problem Reformulation. Our second task is to formulate the problem we have to solve on a given embedding graph. First of all, the R-node subprocedure required by Lemma 13 can be straightforwardly reduced to the task of checking whether a specific shape description ψ can be achieved at the R-node. This reduction takes at most $\mathcal{O}(\tau)$ time by Lemma 4. At this point, the input consists of (1) an R-node μ of T with skeleton H, (2) a mapping $\mathcal{S}_\mu$ which assigns each virtual edge in H to its feasible set, (3) a bound κ on the treewidth of the embedding graph $\widetilde{H}$ obtained from H, and (4) a target shape description ψ.

The combinatorial reformulation we obtain can be stated as follows: Determine if there exists an *angle mapping* α and *shape selector* β which is *valid*, where

- an angle mapping α maps each switch vertex $v \in V(\widetilde{H})$ to a vertex in $N_{\widetilde{H}}(v)$; intuitively, this describes where the large angle at v is in the upward planar embedding of the pertinent graph (this may be in a face between two virtual edges–and α maps v to the corresponding face vertex – or in a virtual edge–and α maps v to that virtual edge),
- a shape selector β maps each edge-vertex v_e obtained from the virtual edge e of H to a shape description that occurs in a feasible set in the range of $\mathcal{S}_\mu(e)$, and
- intuitively, a pair (α, β) is valid if it satisfies three Validity Conditions: (1) all face-vertices receive the correct number of small and large angles from α and β, (2) for each true-vertex v and adjacent edge-vertex w, $\alpha(v)$ is consistent with the requirements of the shape selected by $\beta(w)$, and (3) the shape of the outer face is consistent with ψ.

▶ **Lemma 18.** *There is an upward planar embedding of G_μ with the shape description ψ if and only if there is a valid pair (α, β).*

Finding Valid Pairs Using Treewidth. At this point, what is left to do is solve this combinatorial problem. For the runtime analysis of the algorithm we will develop, we let ζ be the maximum over $n_1(f)$ and $n_{-1}(f)$ (see Theorem 1), over all faces f of all possible planar embeddings of the pertinent graph G_μ of μ. Recalling that no path in G can have length greater than $2^{td(G_\mu)}$ [27], we obtain:

▶ **Observation 19.** *$\zeta \leq V(G_\mu)$, and moreover $\zeta \leq 2^{td(G_\mu)}$.*

We can now design a dynamic program that solves the task at hand. The program computes sets of records for each node of a tree-decomposition in a leaf-to-root fashion, where each record is a tuple of the form $(\texttt{angle}, \texttt{shape}, \texttt{score}, \texttt{left}, \texttt{right})$ where `angle` and `shape` contain snapshots of α and β in the given bag, respectively; `score` keeps track of the sum of large and small angles for each face in the given bag; and `left`, `right` store information about the left-and right-turn-numbers of the outer face.

▶ **Lemma 20.** *There is an algorithm that runs in time $\zeta^{\mathcal{O}(tw(H))} \cdot (|V(H)| + |\mathcal{S}_\mu|)$ and either computes a valid pair, or correctly determines that no such pair exists.*

We now have an R-node subprocedure that runs in XP-time parameterized by treewidth and fixed-parameter time parameterized by treedepth. By invoking Lemma 13, we conclude:

▶ **Theorem 21.** *It is possible to solve* Upward Planarity *in time $n^{\mathcal{O}(tw(G))}$ and time $2^{\mathcal{O}(td(G)^2)} \cdot n^2$, where n is the number of vertices of the input digraph G.*

8 Concluding Remarks

The presented results show that the combination of SPQR-trees with parameterized techniques is a promising algorithmic tool for geometric graph problems. Indeed, for the case of upward planarity, our framework allows us to reduce the general problem to a similar one on 3-connected graphs, at which point it is possible to use parameter-specific approaches such as dynamic programming or flow networks to obtain a solution. We believe not only that the framework developed here can help obtain other algorithms for Upward Planarity, but that the idea behind the framework can be adapted to solve other problems of interest as well – a candidate problem in this regard would be constrained level planarity testing [6].

All algorithms and arguments given within this paper are constructive and can be extended to output an upward planar drawing for each yes-instance of Upward Planarity. An open problem is whether Upward Planarity is W[1]-hard when parameterized by treewidth, or fixed-parameter tractable. Another question is whether the fixed-parameter tractability of Upward Planarity parameterized by the number of sources can be lifted to parameterizing by the maximum turn number of a face in the final drawing.

References

1 Rémy Belmonte, Eun Jung Kim, Michael Lampis, Valia Mitsou, and Yota Otachi. Grundy distinguishes treewidth from pathwidth. In *28th Annual European Symposium on Algorithms, ESA 2020*, volume 173 of *LIPIcs*, pages 14:1–14:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ESA.2020.14.

2 Paola Bertolazzi, Giuseppe Di Battista, Carlo Mannino, and Roberto Tamassia. Optimal upward planarity testing of single-source digraphs. *SIAM J. Comput.*, 27(1):132–169, 1998.

3 Paola Bertolazzi, Giuseppe Di Battista, Giuseppe Liotta, and Carlo Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 12(6):476–497, 1994.

4 Glencora Borradaile, Philip N. Klein, Shay Mozes, Yahav Nussbaum, and Christian Wulff-Nilsen. Multiple-source multiple-sink maximum flow in directed planar graphs in near-linear time. *SIAM J. Comput.*, 46(4):1280–1303, 2017.

5 Guido Brückner, Markus Himmel, and Ignaz Rutter. An SPQR-tree-like embedding representation for upward planarity. In Daniel Archambault and Csaba D. Tóth, editors, *27th International Symposium on Graph Drawing and Network Visualization, GD 2019*, volume 11904 of *Lecture Notes in Computer Science*, pages 517–531. Springer, 2019.

6 Guido Brückner and Ignaz Rutter. Partial and constrained level planarity. In Philip N. Klein, editor, *28th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017*, pages 2000–2011. SIAM, 2017.

7 Hubert Y. Chan. A parameterized algorithm for upward planarity testing. In Susanne Albers and Tomasz Radzik, editors, *12th Annual European Symposium on Algorithms, ESA 2004*, volume 3221 of *Lecture Notes in Computer Science*, pages 157–168. Springer, 2004.

8 Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Robert Ganian, Chrysanthi N. Raftopoulou, and Kirill Simonov. Parameterized algorithms for upward planarity. *CoRR*, abs/2203.05364, 2022. URL: `https://arxiv.org/abs/2203.05364`.

9 Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.

10 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. `doi:10.1007/978-3-319-21275-3`.

11 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.

12 Giuseppe Di Battista and Roberto Tamassia. On-line planarity testing. *SIAM J. Comput.*, 25(5):956–997, 1996.

13 Walter Didimo, Francesco Giordano, and Giuseppe Liotta. Upward spirality and upward planarity testing. *SIAM J. Discret. Math.*, 23(4):1842–1899, 2009.

14 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

15 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. `doi:10.1007/978-1-4471-5559-1`.

16 Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, and Martin Nöllenburg. Extending partial 1-planar drawings. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020*, volume 168 of *LIPIcs*, pages 43:1–43:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

17 Robert Ganian, Thekla Hamm, Fabian Klute, Irene Parada, and Birgit Vogtenhuber. Crossing-optimal extension of simple drawings. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021*, volume 198 of *LIPIcs*, pages 72:1–72:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

18 Robert Ganian, Petr Hlinený, Joachim Kneis, Daniel Meister, Jan Obdrzálek, Peter Rossmanith, and Somnath Sikdar. Are there any good digraph width measures? *J. Comb. Theory, Ser. B*, 116:250–286, 2016.

19 Robert Ganian, Fabrizio Montecchiani, Martin Nöllenburg, and Meirav Zehavi. Parameterized Complexity in Graph Drawing (Dagstuhl Seminar 21293). *Dagstuhl Reports*, 11(6):82–123, 2021.

20 Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. In Roberto Tamassia and Ioannis G. Tollis, editors, *DIMACS International Workshop on Graph Drawing, GD '94*, volume 894 of *Lecture Notes in Computer Science*, pages 286–297. Springer, 1994.

21 Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.

22 Carsten Gutwenger and Petra Mutzel. A linear time implementation of SPQR-trees. In Joe Marks, editor, *8th International Symposium on Graph Drawing, GD '00*, volume 1984 of *Lecture Notes in Computer Science*, pages 77–90. Springer, 2000.

23 Patrick Healy and Karol Lynch. Two fixed-parameter tractable algorithms for testing upward planarity. *Int. J. Found. Comput. Sci.*, 17(5):1095–1114, 2006. doi:10.1142/S0129054106004285.

24 John E. Hopcroft and Robert Endre Tarjan. Dividing a graph into triconnected components. *SIAM J. Comput.*, 2(3):135–158, 1973. doi:10.1137/0202012.

25 Michael D. Hutton and Anna Lubiw. Upward planar drawing of single source acyclic digraphs. In Alok Aggarwal, editor, *2nd Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, SODA 1991*, pages 203–211. ACM/SIAM, 1991.

26 Michael D. Hutton and Anna Lubiw. Upward planar drawing of single-source acyclic digraphs. *SIAM J. Comput.*, 25(2):291–311, 1996. doi:10.1137/S0097539792235906.

27 Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

28 Achilleas Papakostas. Upward planarity testing of outerplanar dags. In Roberto Tamassia and Ioannis G. Tollis, editors, *DIMACS International Workshop on Graph Drawing, GD '94*, volume 894 of *Lecture Notes in Computer Science*, pages 298–306. Springer, 1994. doi:10.1007/3-540-58950-3_385.

29 Neil Robertson and Paul D. Seymour. Graph minors. III. Planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984.

30 William T. Trotter and John I. Moore Jr. The dimension of planar posets. *J. Comb. Theory, Ser. B*, 22(1):54–67, 1977.