Upward Planarity Testing via SAT

Markus Chimani* and Robert Zeranski**

Algorithm Engineering, Friedrich-Schiller-Universität Jena, Germany
{markus.chimani,robert.zeranski}@uni-jena.de

Abstract. A directed acyclic graph is *upward planar* if it allows a drawing without edge crossings where all edges are drawn as curves with monotonously increasing y-coordinates. The problem to decide whether a graph is upward planar or not is NP-complete in general, and while special graph classes are polynomial time solvable, there is not much known about solving the problem for general graphs *in practice*. The only attempt so far was a branch-and-bound algorithm over the graph's triconnectivity structure which was able to solve sparse graphs.

In this paper, we propose a fundamentally different approach, based on the seemingly novel concept of *ordered embeddings*. We carefully model the problem as a special SAT instance, i.e., a logic formula for which we check satisfiability. Solving these SAT instances allows us to decide upward planarity for arbitrary graphs. We then show experimentally that this approach seems to dominate the known alternative approaches and is able to solve traditionally used graph drawing benchmarks effectively.

1 Introduction

A *drawing* of a graph is a mapping of its vertices to points in the plane, and its edges to simple curves connecting the incident vertices but not traversing through the points of non-incident vertices. We say a graph is *planar* if it allows a drawing such that no two edges cross, i.e., no two edge curves traverse a common point other than their edges' common endpoints. For directed graphs, one often considers the special class of *upward* drawings, where we ask for all edges to be drawn as curves monotonously increasing in some specified upward direction (typically the y-axis). In order to allow such a drawing, the graph clearly has to be acyclic, i.e., a *DAG* (directed acyclic graph). There has been a large body of work considering upward drawings, see e.g. [8, 11] for overviews.

Deciding planarity of a graph can be done in linear time [16], but testing whether a DAG allows a planar upward drawing is NP-complete in general [13]. There are, however, several results for special problem variants or graph classes: Upward planarity testing is polynomial when the embedding of the graph (cyclic order of the edges around their incident vertices) is prespecified [2]. Furthermore, bipartite DAGs are upward planar if and only if they are planar [10] and there are polynomial algorithms for outerplanar [18] and series-parallel graphs [11]. In [17], a quadratic testing algorithm—later improved to linear time [2]—has been proposed for single-source DAGs, i.e., graphs

* M. Chimani was funded by a Carl-Zeiss-Foundation juniorprofessorship.
** R. Zeranski was partially supported by the DFG grant MU 1226/8-1 (SPP 1307/2).

W. Didimo and M. Patrignani (Eds.): GD 2012, LNCS 7704, pp. 248–259, 2013.

with a unique vertex without any ingoing edges. Several FPT algorithms were proposed, e.g., by using the number of triconnected components t as their central fixed parameter. The first successful attempt [6] combined this parameter with the number of cut vertices, and was later improved upon to obtain a running time of $O(2^t \cdot t! \cdot n^2)$ [15], where n is the number of vertices. In [11], the factorial factors $t!$ were for the first time substituted by d^t as the most time consuming factors, where d is the diameter of the largest split component. Alternatively, one can obtain an FPT algorithm exponential in the number of edges to remove in order to obtain a tree [15].

Despite those many theoretical results, there are few solutions to tackle the problem in practice. In [3], an (exponential time) algorithm based on branch-and-bound over the possible embeddings was presented, which is able to decide upward planarity in reasonable time for sparse graphs with up to 200 vertices. This lack of practically applicable algorithms is especially unfortunate, as they would not only be interesting for their core functionality, but also to improve different building blocks of other algorithms in the realm of upward drawings: For example, the *upward planarization* approach [7] has to start with a large upward planar subgraph. Due to the lack of fast upward planarity testing algorithms, it uses relatively weak heuristics to find *some* upward planar subgraph which is of reasonable size. We know, however, that good initial subgraphs are a key to good performance in the analogous planarization approach for undirected graphs.

Contribution. We propose a formulation of the upward planarity test as a SAT instance, without any restrictions on the graph structure. It turns out that the problem allows a very concise formulation by using the new concept of *ordered embeddings*. This formulation is very applicable to harness the power of the advanced state-of-the-art SAT solvers. Overall, as we show in our experimental study, this allows us to solve the upward planarity problem in negligible time for graphs very common in graph drawing (with, say, 100 vertices).

In the next section, we introduce the concept of *ordered embeddings*. Section 3 describes the central SAT formulation, while its proof of correctness can be found in Section 4. Finally, Section 5 shows the applicability of our approach.

2 Ordered Embeddings

In the following, we always consider a DAG $G = (V, E)$ for which to test upward planarity. We say an edge e *dominates* an edge f if there is a directed path from e's target to f's source vertex in G. Clearly, f has to be drawn above e in any upward drawing. A pair of edges is *non-dominating* if neither edge dominates the other, and let $\mathcal{N}(G)$ denote the set of all non-dominating edge pairs of G.

A *planar embedding* of a graph is specified by cyclically ordering the edges around their incident vertices, plus the choice of an external face. Most upward planarity algorithms augment this structure by labeling the angles between pairs of cyclically neighboring edges. Herein we propose a different scheme of equivalence classes over all possible upward planar drawings. In fact, our *ordered embeddings* can be seen as finer grained than planar embeddings. Each ordered embedding induces a unique planar embedding, and each planar embedding allows one or more ordered embeddings.

Assume G is upward planar, and consider some upward planar drawing $\mathcal{D}$ of G. Since the drawing is upward planar, it is trivial to perturb the y-coordinates of the vertices slightly such that each vertex of G has a unique y-coordinate. We can then deduce a total order of the vertices V based on their y-coordinates. Consider any pair of edges e, f in the (perturbed) drawing $\mathcal{D}$. We say e and f *overlap* if we can draw a horizontal line L crossing both edges. Based on L we can say whether e is to the left or to the right of f. Observe that in an upward planar drawing, this decision is consistent over all choices for L.

Definition 1 (Partially Ordered Embedding). *A* partially ordered embedding *of a DAG $G = (V, E)$ is a strict total order $<_V$ of the vertices V together with a strict partial order $<_E$ of the edges E. It is feasible if it allows an upward planar drawing such that $u \in V$ is drawn below $v \in V$ iff $u <_V v$ and $e \in E$ is drawn overlapping and to the left of $f \in E$ iff $e <_E f$.*

We are in fact interested in an even stronger embedding variant that is simpler to treat within our SAT approach. Instead of only ordering overlapping edges we want to have a sound order between any two non-dominating edges.

Definition 2 (Fully Ordered Embedding). *A* fully ordered embedding *of a DAG $G = (V, E)$ is a strict total order $<_V$ of the vertices V together with a strict partial order $<^*_E$ of the edges E, where e and f are comparable iff $\{e, f\} \in \mathcal{N}(G)$. It is feasible if it allows an upward planar drawing such that $u \in V$ is drawn below $v \in V$ iff $u <_V v$, and $e \in E$ is drawn to the left (right) of $f \in E$ iff e and f are overlapping and $e <^*_E f$ ($f <^*_E e$, respectively).*

Observe that $<^*_E$ is still only partial, as it does not order dominating edges. Yet, comparability of elements is independent of $<_V$. The *linear extension principle* establishes that any partial order can always be extended to a total order (in fact, for this special setting an independent proof is trivial). This implies:

Proposition 1. *Any feasible partially ordered embedding can be extended to a feasible fully ordered embedding.*

Observe that this extension is not necessarily unique but is guaranteed to include the transitive closure of $<_E$. Considering the inverse direction, any (feasible) fully ordered embedding $(<_V, <^*_E)$ unambiguously induces a sub-relation $<_E$ such that $(<_V, <_E)$ is a (feasible, resp.) partially ordered embedding.

3 SAT Formulation

Again, let $G = (V, E)$ be the DAG for which to decide upward planarity. We will define a logic formula $\mathcal{F}(G)$ in conjunctive normal form (CNF), i.e., we consider a set of *variables*, define a set of *clauses* over them, and ask whether there exists a truth assignment to the variables such that all clauses are satisfied. Canonically each clause is written as a disjunction of (possibly negated) variables. Instead, we give *rules* as

propositional formulae to increase readability. It is straight-forward to transform any such rule into its CNF to obtain the corresponding clauses.

Variables. Our variables should model a fully ordered embedding of G, if it exists. Therefore, we introduce a variable $\tau(v, w)$ for each proper pair of vertices v and w. Intuitively, $\tau(v, w) = \texttt{true}$ means v is *below* w. Since the vertex order is to be strict, $\tau(v, w) = \texttt{false}$ means v is *above* w. For notational simplicity, we may use $\tau(w, v) := \neg\tau(v, w)$ as a shorthand. Furthermore, we introduce variables $\sigma(e, f)$ for each pair $\{e, f\} \in \mathcal{N}(G)$ to model the strict edge order. Thereby, $\sigma(e, f) = \texttt{true}$ means e is *to the left* of f, and the satisfied shorthand $\sigma(f, e) := \neg\sigma(e, f)$ means e is *to the right* of f (if both edges are overlapping).

Rules. As our variable definition already ensures asymmetry, we can ensure strict orders by simply requiring transitivity.

$$\tau(u, v) \wedge \tau(v, w) \to \tau(u, w) \qquad \forall \text{ pairwise distinct } u, v, w \in V \qquad (\mathrm{R}^{\mathrm{t}}_{\tau})$$

$$\sigma(e, f) \wedge \sigma(f, g) \to \sigma(e, g) \qquad \forall \{e, f\}, \{f, g\}, \{e, g\} \in \mathcal{N}(G) \qquad (\mathrm{R}^{\mathrm{t}}_{\sigma})$$

The following *upward rules* ensure that all edges are drawn upward.[1]

$$\tau(v, w) \qquad \forall (v, w) \in E \qquad (\mathrm{R}^{\mathrm{u}})$$

Finally, we require *planarity rules*. Such a rule ensures that two adjacent edges e, f are on the same side of g (non-incident to the common vertex of e and f) if they both overlap with g. Let $e \curvearrowright f$ denote the common vertex of two edges e, f.

$$\Big(\tau(x, u) \wedge \tau(u, y)\Big) \to \Big(\sigma(e, g) \leftrightarrow \sigma(f, g)\Big) \qquad \begin{array}{l} \forall \{e, g\}, \{f, g\} \in \mathcal{N}(G), \\ g = (x, y), e \curvearrowright f = u \notin g. \end{array} \qquad (\mathrm{R}^{\mathrm{p}})$$

ILP formulation. We can easily deduce an integer linear program (ILP) analogous to the SAT formulation. It uses the same variables as binary indicators and has analogous constraints. We will revisit this ILP formulation when discussing the practicability of our approach in Section 5.

4 Validity of the Formulation

We will prove that the above formulation is correct and sufficient:

Theorem 2. *Let $G = (V, E)$ be a DAG. Then, G has an upward planar drawing if and only if $\mathcal{F}(G)$ is satisfiable.*

To prove this theorem, we have to show two directions, formalized as claims:

Claim 3. *An upward planar drawing of G yields a satisfying assignment of $\mathcal{F}(G)$.*

Claim 4. *A satisfying assignment of $\mathcal{F}(G)$ yields an upward planar drawing of G.*

We start with the simpler direction, i.e., Claim 3. Thereafter we prove Claim 4.

[1] Note that these rules are represented by unit-clauses, i.e., $\tau(v, w)$ is fixed to be `true` if the edge (v, w) exists. Via unit propagation (usually done automatically by SAT and ILP solvers) other constraints may become simpler.

4.1 From a Drawing to an Assignment (Claim 3)

Assume that G has an upward planar drawing $\mathcal{D}$. From (perturbed) $\mathcal{D}$, we obtain a feasible partially ordered embedding $(<_V, <_E)$ of G. By Proposition 1 we can extend it to a feasible fully ordered embedding $(<_V, <_E^*)$. We define an assignment $(\bar{\tau}, \bar{\sigma})$ to the τ and σ variables of $\mathcal{F}(G)$ consistent with the intended meaning of the variables:

- $\bar{\tau}(v, w) :=$ `true` if v is below w ($v <_V w$), and $\bar{\tau}(v, w) :=$ `false` otherwise.
- $\bar{\sigma}(e, f) :=$ `true` if e is to the left of f ($e <_E^* f$), and $\bar{\sigma}(e, f) :=$ `false` otherwise.

Notice that $\bar{\tau}$ and $\bar{\sigma}$ represent strict orders. We prove Claim 3 by showing:

Lemma 5. *The assignment $(\bar{\tau}, \bar{\sigma})$ satisfies $\mathcal{F}(G)$.*

Proof. Consider all rules of $\mathcal{F}(G)$ separately. All transitivity rules ($\mathrm{R}^{\mathrm{t}}_\tau$) and ($\mathrm{R}^{\mathrm{t}}_\sigma$) are satisfied by $(\bar{\tau}, \bar{\sigma})$, as $\bar{\tau}$ and $\bar{\sigma}$ are strict orders by definition of the assignment. Furthermore, all upward rules (R^{u}) are satisfied by $\bar{\tau}$, as $\mathcal{D}$ is an upward drawing.

It remains to show that all planarity rules are satisfied. Assume that $(\bar{\tau}, \bar{\sigma})$ violates a planarity rule (R^{p}) for some edge triple e, f, g. Therefore we know $\bar{\tau}(x, u) = \bar{\tau}(u, y) =$ `true` and w.l.o.g. $\bar{\sigma}(e, g) =$ `true` while $\bar{\sigma}(f, g) =$ `false`. Hence in the original ordered embedding we have $e <_E^* g <_E^* f$. Yet, both e and f are overlapping g and in particular, u (the common vertex of e and f) is overlapping with g. It can hence not be on both sides of g at once. □

4.2 From an Assignment to a Drawing (Claim 4)

Let $(\bar{\tau}, \bar{\sigma})$ be a satisfying assignment to $\mathcal{F}(G)$. From it, we can straight-forwardly define a pair of relations using the following transformation $\mathcal{R}$:

- For all defined $\tau(v, w)$: Set $v <_\tau w$ if $\bar{\tau}(v, w) =$ `true`, and $w <_\tau v$ otherwise.
- For all defined $\sigma(e, f)$: Set $e <_\sigma f$ if $\bar{\sigma}(e, f) =$ `true`, and $f <_\sigma e$ otherwise.

Based on the feasibility of the assignments $(\bar{\tau}, \bar{\sigma})$ it is clear:

Observation 6. *The relations $<_\tau$ and $<_\sigma$ are strict. Furthermore, $<_\tau$ is total.*

We now interpret $(<_\tau, <_\sigma)$ as a fully ordered embedding and show that the thereby induced partially ordered embedding $(<_\tau, <_\sigma^p)$ is feasible. To prove this via induction, we first need a technical lemma. Let $H \subset G$ be any subgraph of G. Then $(\bar{\tau}^H, \bar{\sigma}^H)$ and $(<_\tau^H, <_\sigma^H)$ denote the subset of the truth assignment $(\bar{\tau}, \bar{\sigma})$ and the relation system $(<_\tau, <_\sigma)$, respectively, restricted to the vertex pairs and non-dominating edge pairs of H.

Lemma 7. *Let $G = (V, E)$ be a DAG, $(\bar{\tau}, \bar{\sigma})$ a satisfying assignment to $\mathcal{F}(G)$, and $\mathcal{R}(\bar{\tau}, \bar{\sigma}) = (<_\tau, <_\sigma)$ the relation system induced by the assignment, as described above. Let $H \subset G$ be any subgraph of G. We have $\mathcal{R}(\bar{\tau}^H, \bar{\sigma}^H) = (<_\tau^H, <_\sigma^H)$ and feasibility of the ordered embedding $(<_\tau, <_\sigma)$ induces feasibility of $(<_\tau^H, <_\sigma^H)$.*

Proof. Observe that the formula $\mathcal{F}(H)$ is a subformula of $\mathcal{F}(G)$ by definition of $\mathcal{F}$. Hence the sub-assignment $(\bar{\tau}^H, \bar{\sigma}^H)$ is satisfying for $\mathcal{F}(H)$. Consequently, we obtain the same relation system for H independent on whether we apply $\mathcal{R}$ to the reduced truth assignment, or reduce the full relation system $(<_\tau, <_\sigma)$. □

We are now ready to prove Claim 4 via the following lemma:

Lemma 8. *Let $G = (V, E)$ be a DAG, $(\bar{\tau}, \bar{\sigma})$ a satisfying assignment to $\mathcal{F}(G)$, $\mathcal{R}(\bar{\tau}, \bar{\sigma}) = (<_\tau, <_\sigma)$ the induced relation system, and $<_\sigma^p$ be the sub-relation of $<_\sigma$ such that $(<_\tau, <_\sigma^p)$ is a partially ordered embedding. This embedding is feasible.*

Proof. The proof is by induction on the number of edges of G. Therefore, we will label the edges of $G = (V, E)$ as $\{e_1, \ldots, e_m\}$ according to the following scheme (not to be confused with $<_\sigma$). The definition is incremental for increasing $i = 1, \ldots, m$. Let $C_i := E \setminus \{e_1, \ldots, e_{i-1}\}$ (with $C_1 := E$) denote the edges that have not received an index less than i, and are thus candidates for e_i.

$$e_i := \max_{<_\sigma} \Big\{ (u, v) \in C_i : \nexists (x, y) \in C_i \text{ with } u <_\tau x \Big\}$$

Intuitively, we order the edges according to their source vertex from high to low. Multiple edges emanating from the same source vertex are ordered from right to left. Using the edge labels, we define $E_i := \{e_1, \ldots, e_i\}$ as the set of edges with index at most i, and $G_i := G[E_i]$ as the graph induced by the first i edges.

Induction hypothesis and base case. Given any relation system $(<_\tau^{G_i}, <_\sigma^{G_i})$ restricted to some subgraph $G_i \subset G$, let $<_\sigma^{G_i,p}$ denote the sub-relation of $<_\sigma^{G_i}$ such that $(<_\tau^{G_i}, <_\sigma^{G_i,p})$ is a partially ordered embedding of G_i. As our induction hypothesis, we assume that this embedding is feasible. This trivially holds for $i = 1$ since G_1 is a single edge being drawn upward.

Induction step. We have to show that $(<_\tau^{G_{i+1}}, <_\sigma^{G_{i+1},p})$ is a feasible partially ordered embedding. By our induction hypothesis $(<_\tau^{G_i}, <_\sigma^{G_i,p})$ is feasible and by Lemma 7 these relations are sub-relations of $<_\tau^{G_{i+1}}$ and $<_\sigma^{G_{i+1},p}$, respectively. We have to show that e_{i+1} can be drawn, consistent to $(<_\tau^{G_{i+1}}, <_\sigma^{G_{i+1},p})$, into a drawing realizing $(<_\tau^{G_i}, <_\sigma^{G_i,p})$. The reduction to partially ordered embeddings allows us to restrict our attention to edge relations between overlapping edges.

> **Claim 9.** *Let $G = (V, E)$ be a DAG, $(<_\tau^{G_i}, <_\sigma^{G_i,p})$ a feasible partially ordered embedding of G_i, and $\mathcal{D}_i$ a drawing of G_i satisfying this embedding. We can draw the edge e_{i+1} into $\mathcal{D}_i$ without crossings and consistent with the relation system $(<_\tau^{G_{i+1}}, <_\sigma^{G_{i+1},p})$.*

Proving this claim will establish the lemma. As the claim's proof itself is based on induction, we give the proof separately below to avoid confusion. □

Proof of Claim 9. Consider the vertices in $\mathcal{D}_i$ together with the possibly additional vertices incident to e_{i+1}. Label them $v_1, \ldots, v_\ell$ from bottom to top according to $<_\tau^{G_{i+1}}$. This order is consistent with $<_\tau^{G_i}$ and hence with $\mathcal{D}_i$. By construction of our edge

labels $e_1, \ldots, e_m$, the lowest vertex v_1 is e_{i+1}'s source vertex. Let v_k denote e_{i+1}'s target vertex which may or may not be already drawn in $\mathcal{D}_i$. We can observe a central property at any vertex v_j with $1 < j < k$:

(P) Let A_j be the edges incident to v_j, $1 < j < k$. Either all edges of A_j are to the left, or all those edges are to the right of e_{i+1}.

This property follows from the planarity rule (R^{p}) with $g := e_{i+1}$ and $e, f \in A_j$. The left side of the implication is true since $\bar{\tau}(v_1, v_j) = \bar{\tau}(v_j, v_k) = \texttt{true}$. Therefore, $\sigma(e, e_{i+1}) \leftrightarrow \sigma(f, e_{i+1})$ holds for all edge pairs $e, f \in A_j$. Since A_j is always non-empty in an edge-induced graph G_i, we may say v_j is to the left (right) of e_{i+1} if at least one edge of A_j is to its left (right, respectively).

In $\mathcal{D}_i$ we can think of a *layer* j as the hypothetical horizontal line through vertex v_j. We will construct a consistent drawing of e_{i+1} into $\mathcal{D}_i$ by induction on the number of layers the edge's drawing traverses. Intuitively, we will draw the edge e_{i+1} piecewise from layer to layer, consistent to the overlapped edges, i.e., $<_\sigma^{G_{i+1},p}$. We can assume w.l.o.g. that each vertex v_j in $\mathcal{D}_i$ is drawn at y-coordinate j. Sweeping a horizontal line from the bottom of the drawing upwards, it is clear that the sweep-line hits new edges only at integer y-coordinates.

Induction hypothesis and base case. An edge (v_a, v_b) *region-overlaps* with an edge $(v_{a'}, v_{b'})$ at region j iff $a, a' \leq j$ and $b, b' \geq j + 1$. Hence e_{i+1} overlaps with another edge f iff they region-overlap at some region j, $s \leq j < k$. As our induction hypothesis let e_{i+1} be drawn in an upward fashion to slightly above layer j (i.e., up to y-coordinate $j + \varepsilon$ for some small $\varepsilon > 0$), such that it satisfies the relation $<_\sigma^{G_{i+1},p}$ for all edges overlapping with e at any region $j' \leq j$.

For the base case $j = 1$, e_{i+1} is the left-most edge of all edges leaving v_1 in G_{i+1}, by construction of the edge labels $e_1, \ldots, e_m$. Since the graph contains no vertex below v_1, we can draw the first small piece of e_{i+1} left of all other edges A_1 from v_1 to y-coordinate $1 + \varepsilon$ without any crossings while satisfying $<_\sigma^{G_{i+1},p}$.

Induction step. We perform the induction only for $j + 1 < k$ and extend the drawing of e_{i+1} from y-coordinate $j + \varepsilon$ to $j + 1 + \varepsilon$. Let $F_j := \{(v_a, v_b) \in E_i : a \leq j, b \geq j + 1\}$ be the edges traversing the region between the layers j and $j + 1$. Since they all overlap with each other and with e_{i+1}, the relation $<_\sigma^{G_{i+1},p}$ when restricted to edge pairs of F_j is a total strict order. This allows us to consider the direct successor $f \in F_j$ and direct predecessor $f' \in F_j$ of e_{i+1} (if they exist) w.r.t. this total order. For e_{i+1}, we will choosing a routing similar to one of these neighbors (if they exist). If v_{j+1} is to the left of e_{i+1} (cf. property (P)), we pick the successor f; otherwise, we pick f'.

As the two cases are symmetric, assume w.l.o.g. that v_{j+1} is to the left of e_{i+1} and hence we pick f. Assume f does not exist then there is no edge to the right of e_{i+1} in the y-coordinate interval $[j, j + 1)$ (recall that new edges may only appear at coordinate $j + 1$). Hence we can draw the next segment of e_{i+1} easily to the right of the whole drawing of $\mathcal{D}_i$ within this interval. Since all new edges at y-coordinate $j + 1$ emanate from v_{j+1} (which is to the left of e_{i+1}), we can extend the drawing of e_{i+1} to cross layer $j + 1$ and go up to y-coordinate $j + 1 + \varepsilon$ without introducing any crossings while staying true to $<_\sigma^{G_{i+1},p}$.

Assume f exists. Since f is the direct successor of e_{i+1} in F_j we can extend the drawing of e_{i+1} from its currently last drawn point into the direct vicinity of f without introducing crossings. By our choice of f over f' we know that v_{j+1} is not the target vertex of f and moreover (by transitivity) we know that f is to the right of v_{j+1}. Hence, we route e_{i+1} close to f (without touching or crossing it) up to layer $j+1$ and a bit beyond (up to y-coordinate $j+1+\varepsilon$). There arise no crossings from this operation. As e_{i+1}'s drawing was consistent with $<_\sigma^{G_{i+1},p}$ at y-coordinate $j+\varepsilon$ by our induction hypothesis, it stays so for all region-overlapping edges F_j. Furthermore, it is consistent with all edges having v_{j+1} as their source vertex. Hence it is consistent with all region-overlapping edges for regions $j' \leq j+1$.

Final step. Consider the case $j = k-1$. By our induction hypothesis we have a drawing of e_{i+1} up to y-coordinate $k-1+\varepsilon$. Define F_j as above and consider the strict total order over $F_j \cup \{e_{i+1}\}$ induced by $<_\sigma^{G_{i+1},p}$. If e_{i+1}'s direct predecessor or successor w.r.t. this order has target vertex v_k, bring e_{i+1} close to this edge's drawing and draw e_{i+1} close to it up to vertex v_k without any crossings. Otherwise, pick successor f. Assume f exists and recall that e_{i+1} is incident to v_k. Property (P) guarantees that f is the closest edge to the right of v_k. Hence we extend the drawing of e_{i+1} by first bringing it close to f, routing along side it up to y-coordinate $k-\varepsilon$. If v_k is already contained in $\mathcal{D}_i$, we can finish the drawing by going directly to it without crossings. If $v_k \notin \mathcal{D}_i$, we can simply extend the drawing to y-coordinate $k-\varepsilon$ without further crossings and place v_k at this end point.

If f does not exist, there are no edges to the right of e_{i+1} in the y-coordinate interval $[k-1, k)$ and we can easily draw the final edge segment to the right of $\mathcal{D}_i$ in this interval, ending at vertex v_k (possibly introducing the vertex into the drawing to the right of the rest of the drawing) without any crossings. Since we never introduced crossings and the drawing before the extension was already consistent with all overlapping edges, this closes the proof of Claim 9—and consequently of Lemma 8, Claim 4, and thus Theorem 2. □

5 Experiments

All our experiments were performed on an Intel Xeon E5520, 2.27 GHz, 8 GB RAM running Debian 6. We implemented our algorithms in the Open Graph Drawing Framework (OGDF, `www.ogdf.net`), using *minisat* as our SAT solver and CPLEX 12 as our ILP solver. For both solvers, we used their default settings.

Established Benchmark Sets. We start with analyzing the behavior of our SAT formulation on established benchmark sets. The *Rome* graphs [19] are 11528 undirected graphs with up to 100 vertices and 158 edges. We directed the graphs canonically, as proposed in [12], to obtain DAGs. The *North DAGs* [9] are 1277 graphs, collected by Stephen North, were originally proposed to compare algorithms for drawing DAGs. Since planarity of the underlying undirected graph is necessary for upward planarity (and efficiently testable), we restrict our attention to the 3281 and 854 planar graphs of those benchmark sets, respectively.

Table 1. Number of solved instances (Rome and North benchmark set) within t seconds for the SAT and the ILP approach. We applied a timeout of 1 minute. *) Both instances require less than 0.2 seconds.

		$t \leq 0.01$	$0.01 < t \leq 0.1$	$0.1 < t \leq 1$	$1 < t \leq 10$	$10 < t \leq 60$	$t > 60$
Rome	SAT	2055	1132	85	7	2	0
	ILP	148	793	479	341	286	1235
North	SAT	814	38	2 *	0	0	0
	ILP	174	215	190	88	55	132

Table 2. Number of upward planar instances (out of the Rome and North benchmark set) requiring c crossing when computing LFUP

	overall	$c = 0$	$c = 1$	$c = 2$	$c = 3$	$c = 4$	$c = 5$	$c = 6$	$c = 7$	$c = 8$	$c = 9$	$c \geq 10$
Rome	3092	1734	535	307	179	125	72	42	38	19	14	27
North	813	746	23	19	8	5	2	3	2	2	0	3

For each instance, we apply simple a preprocessing strategy to shrink the input graph without effecting its property w.r.t. upward-planarity: We iteratively delete all degree-1 vertices, and replace directed paths between any two vertices s and t where all inner vertices have degree 2 by a single edge (s, t).

Runtime. We first consider the runtime performance of our SAT approach, and can compare it to the analogous ILP formulation sketched at the end of Section 3.

The SAT approach proves to be very efficient in practice, solving 97,1% (62.6%) of all Rome graphs within 0.1 seconds (0.01 seconds), see Table 1. In contrast to this, the ILP approach solves only 28.7% (4.5%) of the instances within these time bounds. Every Rome graph can be solved within 45 seconds by SAT, whereas the ILP variant is not able to solve 39.2% of the graphs within this bound. The results for the North DAGs are similar: All instances are solved by SAT within 0.2 seconds (95.3% within 0.01 seconds), whereas the ILP approach solves only 56.7% (20.3%) within these time bounds. The ILP approach cannot finish over 15.5% of the instances within 60 seconds. We conclude that the SAT approach is very effective in practice, while the analogous ILP is clearly inferior: We observe that the solutions to the LP relaxations are highly fractional, and thus the ILP solver has to branch very often. SAT-solvers, on the other hand, are especially designed for extensive branching.

Crossing minimization. When drawing DAGs in practice, one usually tries to minimize crossings. For sparse graphs (thus including the Rome and North instance), the *Layer Free Upward Planarization* method (LFUP) [7] is known to be the currently strongest approach [1]. Using our new tool, we can ask how often LFUP achieves the optimum upward crossing number of 0 for upward planar graphs (Table 2). LFUP works well for the North DAGs, as it achieves 0 crossings for 91.8% of the upward planar graphs. Interestingly, it performs worse for the Rome graphs, where it achieves the optimal solution only for 56.1% of the upward planar instances, and requires more than 3 crossings for 10.8%.

Bend minimization. As mentioned earlier, Bertalozzi et al. [3] proposed a sophisticated branch-and-bound strategy over the graph's triconnectivity structure to minimize the

number of *bends* in quasi-upward planar drawings. Thereby, upward planar graphs are exactly those requiring 0 bends. Hence, this tool can also be used to test upward planarity. Unfortunately, however, we are unable to make a fully fair comparison between both approaches: the original code is unavailable by now[2], and the paper only reports on the full running times, not restricted to only deciding upward planarity. Furthermore, the experiments were conducted on a by now outdated machine (Sun UltraSPARC I)[3] and only 200 out of the originally 300 used benchmark instances (random graphs with up to 200 vertices) are still available[2]. Out of those graphs, many do contain directed cycles (and are hence trivially non-upward planar). For our algorithm, we consider the 61 instances that are DAGs. The running time (over all 300 instances) of the branch-and-bound approach was summarized as requiring "an average time that is less than 20 minutes for digraphs with a high number of vertices" [3], and the diagram suggests 10–20 minutes for the graphs with 150–200 vertices. Using our SAT approach, we require 0.36 seconds on average (92% under one second), over the 61 DAG instances.

Generated Graphs. To further investigate the behavior of our SAT approach, we generated random graphs with $n \in \{100, 150, 200, 250\}$ vertices (6 graphs for each n). Each graph is generated as follows: Starting with an empty graph, we pick an edge not yet in the graph uniformly at random. We only pick edges such that the underlying graph remains planar and choose an orientation of the edge such that the graph remains a DAG. Let G be the generated (planar, triangulated) DAG, then G_i denotes the subgraph containing the first i edges according to the generation process. For every graph G there is a z such that G_z is upward planar but G_{z+1} is not—these can be considered to be instances specifically constructed to be hard. We are interested in this phase transition, see Figure 1(left). As we would expect, the running time increases for $G_1, \ldots, G_z$ and decreases afterwards. At some point afterwards, the running time is mainly dominated by the parsing time. Hence upward planarity testing is particularly fast for graphs sufficiently "far away" from the phase transition; it is fastest for sparse, upward planar graphs and for dense non-upward planar graphs. This agrees with our results in Section 5, as both the Rome and North benchmark sets contain mainly sparse graphs: Their average running time for upward planar instances is 0.01 seconds, while it is 0.2 seconds for the non-upward planar instances.

Finally, we investigate the algorithm's dependence on the number of sources and sinks (recall that the problem is polynomial if there is a single source). We generated random graphs with a prespecified number of s sources and t sinks: We start with a random tree spanning all n vertices with $s + t$ leaves, and direct the edges naturally from s source-leaves to t sink-leaves. Then we continue to grow our graph as before, ensuring that no sources and sinks get destroyed. We generated graphs with 150 vertices and 1/1, 5/5, 10/10, 20/20 sources/sinks (4 each). Our experiments show that the running time increases with increasing number of sources and sinks, cf. Figure 1(right), but even the hardest instances are always well below 15 seconds. We also observe that for more sources and sinks, the hardest instances are typically the ones having a couple of edges too many to be upward planar.

[2] Personal communication.

[3] SPECint benchmark results suggest that our machine is roughly $22 - 36$ times faster.

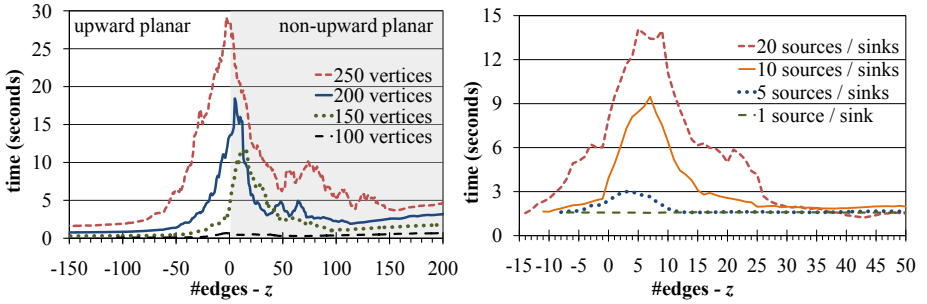

Fig. 1. Running time of the SAT approach for our generated graphs of varying size (left) and for those with prespecified number of sources and sinks (150 vertices) (right)

6 Conclusion

We introduced the concept of *ordered embeddings* and showed how to use them to devise a conceptually surprisingly elegant SAT formulation to solve the upward planarity problem for general graphs. Our experiments show that this approach is very applicable in practice, and dominates both the previously known algorithm and the ILP formulation based on the same concepts as the SAT formulation.

There are some straight-forward extensions to our model. In many practical scenarios, one is interested in certain *embedding constraints*, as formalized in [14], where we may restrict the order of incident edges around a vertex. We can straight-forwardly introduce such restrictions into our SAT formulation.

Another generalization that gained some interest lately are *mixed graphs*, where some edges are undirected and we ask whether they can be oriented such that the resulting graph becomes upward planar [4, 5]. By dropping the upward rules (R^u) for undirected edges, our formulation will solve this problem. In [5], the generally simpler special case of a given fixed embedding is considered. While our formulation could also be extended to this fixed embedding case, the problem allows for more direct and practically more efficient ILP formulations.

Overall, having our new tools at hand, they could allow us to devise practically more efficient schemes for related problems as well, e.g., when computing large feasible upward planar subgraphs within the upward planarization approach. It could also be worthwhile to extend the above model to PBS (pseudo boolean satisfiability) models in order to solve optimization problems as maximum upward planar subgraph or minimum upward crossing number.

Acknowledgements. We thank Walter Didimo for kindly providing information, preprints, and benchmark instances.

References

1. Bachmaier, C., Brunner, W., Gleißner, A.: Grid sifting: Leveling and crossing reduction. Tech. Rep. MIP-1103, University of Passau (2011)
2. Bertolazzi, P., Di Battista, G., Liotta, G., Mannino, C.: Upward drawings of triconnected digraphs. Algorithmica 12(6), 476–497 (1994)

3. Bertolazzi, P., Di Battista, G., Didimo, W.: Quasi-upward planarity. Algorithmica 32(3), 474–506 (2002)
4. Binucci, C., Didimo, W.: Upward Planarity Testing of Embedded Mixed Graphs. In: van Kreveld, M., Speckmann, B. (eds.) GD 2011. LNCS, vol. 7034, pp. 427–432. Springer, Heidelberg (2012)
5. Binucci, C., Didimo, W., Patrignani, M.: Upward and quasi-upward planarity testing of embedded mixed graphs. Tech. Rep. RT-001-12, University of Perugia, Department of Electronic and Information Engineering (2012)
6. Chan, H.: A Parameterized Algorithm for Upward Planarity Testing. In: Albers, S., Radzik, T. (eds.) ESA 2004. LNCS, vol. 3221, pp. 157–168. Springer, Heidelberg (2004)
7. Chimani, M., Gutwenger, C., Mutzel, P., Wong, H.M.: Layer-free upward crossing minimization. ACM Journal of Experimental Algorithmics 15 (2010)
8. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing. Prentice Hall, Upper Saddle River (1999)
9. Di Battista, G., Garg, A., Liotta, G., Parise, A., Tamassia, R., Tassinari, E., Vargiu, F., Vismara, L.: Drawing directed acyclic graphs: An experimental study. International Journal of Computational Geometry and Appl. 10(6), 623–648 (2000)
10. Di Battista, G., Liu, W.P., Rival, I.: Bipartite graphs, upward drawings, and planarity. Information Processing Letters 36(6), 317–322 (1990)
11. Didimo, W., Giordano, F., Liotta, G.: Upward spirality and upward planarity testing. SIAM Journal on Discrete Mathematics 23(4), 1842–1899
12. Eiglsperger, M., Kaufmann, M.: An Approach for Mixed Upward Planarization. In: Dehne, F., Sack, J.-R., Tamassia, R. (eds.) WADS 2001. LNCS, vol. 2125, pp. 352–364. Springer, Heidelberg (2001)
13. Garg, A., Tamassia, R.: On the computational complexity of upward and rectilinear planarity testing. SIAM Journal on Computing 31(2), 601–625 (2002)
14. Gutwenger, C., Klein, K., Mutzel, P.: Planarity testing and optimal edge insertion with embedding constraints. J. Graph Algorithms Appl. 12(1), 73–95 (2008)
15. Healy, P., Lynch, K.: Fixed-Parameter Tractable Algorithms for Testing Upward Planarity. In: Vojtáš, P., Bieliková, M., Charron-Bost, B., Sýkora, O. (eds.) SOFSEM 2005. LNCS, vol. 3381, pp. 199–208. Springer, Heidelberg (2005)
16. Hopcroft, J., Tarjan, R.: Efficient planarity testing. J. ACM 21(4), 549–568 (1974)
17. Hutton, M.D., Lubiw, A.: Upward planar drawing of single source acyclic digraphs. In: Proc. of SODA 1991, pp. 203–211 (1991)
18. Papakostas, A.: Upward Planarity Testing of Outerplanar Dags (Extended Abstract). In: Tamassia, R., Tollis, I.G. (eds.) GD 1994. LNCS, vol. 894, pp. 298–306. Springer, Heidelberg (1995)
19. Welzl, E., Di Battista, G., Garg, A., Liotta, G., Tamassia, R., Tassinari, E., Vargiu, F.: An experimental comparison of four graph drawing algorithms. Computational Geometry 7, 303–325 (1997)