A Universal Point Set for 2-Outerplanar Graphs*

Patrizio Angelini[1], Till Bruckdorfer[1], Michael Kaufmann[1], and Tamara Mchedlidze[2]

[1] Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, Germany
[2] Institute of Theoretical Informatics, Karlsruhe Institute of Technology, Germany

Abstract. A point set $S \subseteq \mathbb{R}^2$ is universal for a class $\mathcal{G}$ if every graph of $\mathcal{G}$ has a planar straight-line embedding on S. It is well-known that the integer grid is a quadratic-size universal point set for planar graphs, while the existence of a sub-quadratic universal point set for them is one of the most fascinating open problems in Graph Drawing. Motivated by the fact that outerplanarity is a key property for the existence of small universal point sets, we study 2-outerplanar graphs and provide for them a universal point set of size $O(n \log n)$.

1 Introduction

Let S be a set of m points on the plane. A *planar straight-line embedding* of an n-vertex planar graph G, with $n \leq m$, on S is a mapping of each vertex of G to a distinct point of S so that, if the edges are drawn straight-line, no two edges cross. Point set S is *universal* for a class $\mathcal{G}$ of graphs if every graph $G \in \mathcal{G}$ has a planar straight-line embedding on S. Asymptotically, the smallest universal point set for general planar graphs is known to have size at least $1.235n$ [11], while the upper bound is $O(n^2)$ [3, 8, 12]. All the upper bounds are based on drawing the graphs on an integer grid, except for the one by Bannister et al. [3], who use super-patterns to obtain a universal point set of size $n^2/4 - \Theta(n)$ – currently the best result for planar graphs. Closing the gap between the lower and the upper bounds is a challenging open problem [6–8].

A subclass of planar graphs for which the "smallest possible" universal point set is known is the class of *outerplanar* graphs – the graphs that admit a straight-line planar drawing in which all vertices are incident to the outer face. Namely, Gritzmann et al. [10] and Bose [5] proved that any size-n point set in general position is universal for n-vertex outerplanar graphs. Motivated by this result, we consider the class of k-*outerplanar* graphs, with $k \geq 2$, which is a generalization of outerplanar graphs. A planar drawing of a graph is k-outerplanar if removing the vertices of the outer face, called k-th *level*, produces a $(k-1)$-outerplanar drawing, where 1-outerplanar stands for outerplanar. A graph is k-outerplanar if it admits a k-outerplanar drawing. Note that every planar graph is a k-outerplanar graph, for some value of $k \in O(n)$. Hence, in order to tackle a meaningful subproblem of the general one, it makes sense to study the existence of subquadratic universal point sets when the value of k is bounded by a constant or a sublinear function. However, while the case $k = 1$ is trivially solved by selecting any n points in general position, as observed above [5, 10], the case $k = 2$ already

* This work has been supported by DFG grant Ka812/17-1. The full version of the paper, including all the missing proofs, can be found in [2].

eluded several attempts of solution and turned out to be far from trivial. In this paper, we finally solve the case $k = 2$ by providing a universal point set for 2-outerplanar graphs of size $O(n \log n)$.

A subclass of k-outerplanar graphs, in which the value of k is unbounded, but every level is restricted to be a chordless simple cycle, was known to have a universal point set of size $O(n(\frac{\log n}{\log \log n})^2)$ [1], which was subsequently reduced to $O(n \log n)$ [3]. It is also known that *planar 3-trees* – graphs not defined in terms of k-outerplanarity – have a universal point set of size $O(n^{5/3})$ [9]. Note that planar 3-trees have treewidth equal to 3, while 2-outerplanar graphs have treewidth at most 5.

Structure of the paper: After some preliminaries and definitions in Section 2, we consider 2-outerplanar graphs in Section 3 where the inner level is a forest and all the internal faces are triangles. We prove that this class of graphs admits a universal point set of size $O(n^{3/2})$. We then extend the result in Section 4 to 2-outerplanar graphs in which the inner level is still a forest but the faces are allowed to have larger size. Finally, in Section 5, we outline how the result of Section 4 can be extended to general 2-outerplanar graphs. We also explain how to apply the methods in [3] to reduce the size of the point set to $O(n \log n)$. We conclude with open problems in Section 6.

2 Preliminaries and Definitions

A straight-line segment with endpoints p and q is denoted by $s(pq)$. A circular arc with endpoints p and q (clockwise) is denoted by $a(pq)$. We assume familiarity with the concepts of *planar graphs*, *straight-line planar drawings* and their *faces*. A straight-line planar drawing Γ of a graph G determines a clockwise ordering of the edges incident to each vertex u of G, called *rotation at* u. The *rotation scheme* of G in Γ is the set of the rotations at all the vertices of G determined by Γ. Observe that, if G is connected, in all the straight-line planar drawings of G determining the same rotation scheme, the faces of the drawing are delimited by the same edges.

Let $[G, \mathcal{H}]$ be a 2-outerplanar graph, where the outer level is an outerplanar graph G and the inner level is a set $\mathcal{H} = \{G_1, \ldots, G_k\}$ of outerplanar graphs. We assume that $[G, \mathcal{H}]$ is given together with a rotation scheme, and the goal is to construct a planar straight-line embedding of $[G, \mathcal{H}]$ on a point set determining this rotation scheme. Since $[G, \mathcal{H}]$ can be assumed to be connected (as otherwise we can add a minimal set of dummy edges to make it connected), this is equivalent to assuming that a straight-line planar drawing Γ of $[G, \mathcal{H}]$ is given. We rename the faces of Γ as $F_1, \ldots, F_k$ in such a way that each graph G_h, which can also be assumed connected, lies inside face F_h. Note that, for each face F_h of G, the graph $[F_h, G_h]$ is again a 2-outerplanar graph; however, its outer level F_h is a simple chordless cycle and its inner level G_h consists of only one connected component. In the special case in which G_h is a tree we say that graph $[F_h, G_h]$ is a *cycle-tree* graph. We say that a 2-outerplanar graph is *inner-triangulated* if all the internal faces are 3-cycles. Note that not every cycle-tree graph can be augmented to be inner-triangulated without introducing multiple edges.

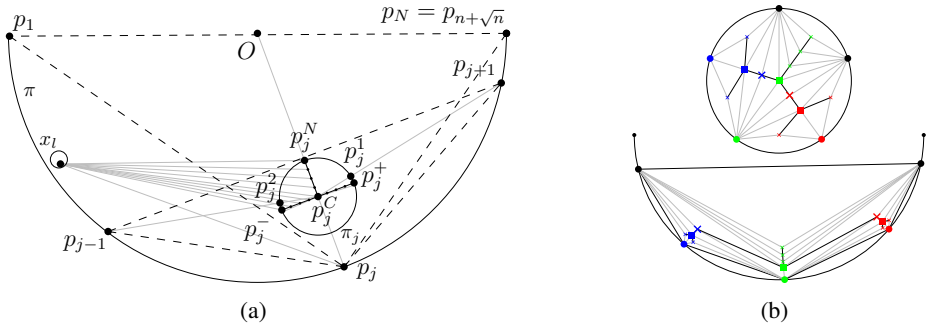


Fig. 1. (a) Illustration of S, focused on S_j of p_j. (b) A cycle-tree graph and its embedding.

3 Inner-Triangulated 2-Outerplanar Graphs with Forest

In this section we prove that there exists a universal point set S of size $O(n^{3/2})$ for the class of n-vertex inner-triangulated 2-outerplanar graphs $[G, \mathcal{H}]$ where $\mathcal{H}$ is a forest.

3.1 Construction of the Universal Point Set

In the following we describe S (Fig. 1(a)). Let π be a half circle with center O and let $N := n + \sqrt{n}$. Uniformly distribute points in $S_{\mathcal{M}} = \{p_1, \ldots, p_N\}$ on π. The points in $S_{\mathcal{D}} = \{p_{i\sqrt{n}+i} : 1 \leq i \leq \sqrt{n}\}$ are called *dense*, while the remaining points in $S_{\mathcal{M}} \setminus S_{\mathcal{D}}$ are *sparse*[3]. For $j = 2, \ldots, N-1$, place a circle π_j with its center p_j^C on $s(p_jO)$, so that it lies completely inside the triangle $\triangle p_{j-1}p_jp_{j+1}$ and inside the triangle $\triangle p_1p_jp_N$. Note that the angles $\angle p_jp_j^Cp_N$ and $\angle p_jp_j^Cp_1$ are smaller than $180°$. Let p_j^N be the intersection point between $s(p_jO)$ and π_j that is closer to O. Also, let p_j^1 (resp. p_j^2) be the intersection point of $s(p_j^Cp_{j+1})$ (resp. $s(p_j^Cp_{j-1})$) with π_j. Finally, let p_j^3 (resp. p_j^4) be the intersection point of π_j with its diameter orthogonal to $s(p_jO)$, such that $a(p_j^3p_j^4)$ does not contain p_j^N. Now, choose a point p_j^+ on the arc $a(p_j^1p_j^3)$, and a point p_j^- on the arc $a(p_j^4p_j^2)$. To complete the construction of S, evenly distribute $\overline{n} - 1$ points on each of the three segments $s_j^N := s(p_j^Cp_j^N)$, $s_j^+ := s(p_j^Cp_j^+)$, and $s_j^- := s(p_j^Cp_j^-)$, where $\overline{n} = n$ if p_j is dense and $\overline{n} = \sqrt{n}$ if it is sparse. We refer to the points on s^N, s^+, s^-, including the points $p_j^N, p_j^C, p_j^+, p_j^-$, as *the point set of* p_j, and we denote it by S_j. Vertex p_j^C is the *center vertex of* S_j. The described construction uses $O(n^{3/2})$ points and ensures the following property.

Property 1. For each $j = 1, \ldots, N$, the following visibility properties hold:
(A) The straight-line segments connecting point p_j to: point p_j^-, to the points on s_j^-, to p_j^C, to the points on s_j^+, and to p_j^+ appear in this clockwise order around p_j.
(B) For all $l < j$, consider any point $x_l \in \{p_l\} \cup S_l$ (see Fig. 1(a)); then, the straight-line segments connecting x_l to: p_j^N, to the points on s_j^N, to p_j^C, to the points on s_j^-, to

[3] The distribution of the points into dense and sparse portions of the point set is inspired by [1].

p_j^-, and to p_j appear in this clockwise order around x_l. Also, consider the line passing through x_l and any point in $\{p_j\} \cup S_j$; then, every point in $\{p_q\} \cup S_q$, with $l < q < j$, lies in the half-plane delimited by this line that does not contain the center O of π.
(C) For all $l > j$, consider any point $x_l \in \{p_l\} \cup S_l$; then, the straight-line segments connecting x_l to: p_j^N, to the points on s_j^N, to p_j^C, to the points on s_j^+, to p_j^+, and to p_j appear in this counterclockwise order around x_l. Also, consider the line passing through x_l and any point in $\{p_j\} \cup S_j$; then, every point in $\{p_q\} \cup S_q$, with $j < q < l$, lies in the half-plane delimited by this line that does not contain O.

3.2 Labeling the Graph

Let $[G, \mathcal{H}]$ be an inner-triangulated 2-outerplanar graph where G is an outerplanar graph and $\mathcal{H} = \{T_1, \ldots, T_k\}$ is a forest such that tree T_h lies inside face F_h of G, for each $1 \leq h \leq k$. The idea behind the labeling is the following: in our embedding strategy, G will be embedded on the half-circle π of the point set S, while the tree $T_h \in \mathcal{H}$ lying inside each face F_h of G will be embedded on the point sets S_j of some of the points p_j on which vertices of F_h are placed. Note that, since π is a half-circle, the drawing of F_h will always be a convex polygon in which two vertices have *small* (acute) internal angles, while all the other vertices have *large* (obtuse) internal angles. In particular, the vertices with the small angle are the first and the last vertices of F_h in the order in which they appear along the outer face of Γ. Since, by construction, a point p_j of F_h has its point set S_j in the interior of F_h if and only if it has a large angle, we aim at assigning each vertex of T_h to a vertex of F_h that is neither the first nor the last. We will describe this assignment by means of a labeling $\ell\colon [G, \mathcal{H}] \to 1, \ldots, |G|$; namely, we will assign a distinct label $\ell(v)$ to each vertex $v \in G$ and then assign to each vertex of T_h the same label as one of the vertices of F_h that is neither the first or the last. Then, the number of vertices with the same label as a vertex of G will determine whether this vertex will be placed on a sparse or a dense point. We formalize this idea in the following.

We rename the vertices of G as $v_1, \ldots, v_{|G|}$ in the order in which they appear along the outer face of Γ, and label them with $\ell(v_i) = i$ for $i = 1, \ldots, |G|$. Next, we label the vertices of each tree $T_h \in \mathcal{H}$. Since trees T_h and $T_{h'}$ are disjoint for $h \neq h'$, we focus on the cycle-tree graph $[F, T]$ composed of a single face $F = F_h$ of G and of the tree $T = T_h \in \mathcal{H}$ inside it. Rename the vertices of F as $w_1, \ldots, w_m$ in such a way that for any two vertices $w_x = v_p$ and $w_{x+1} = v_q$, where $p, q \in \{1, \ldots, |G|\}$, it holds that $p < q$. As a result, w_1 and w_m are the only vertices of F with small internal angles. A vertex of T is a *fork vertex* if it is adjacent to more than two vertices of F (square vertices in Fig. 1(b)), otherwise it is a *non-fork vertex* (cross vertices in Fig. 1(b)). Since $[F, T]$ is inner-triangulated, every vertex of T is adjacent to at least two vertices of F, and hence non-fork vertices are adjacent to exactly two vertices of F. We label the vertices of T starting from its fork vertices. To this end, we construct a tree T' composed only of the fork vertices, as follows. Initialize T'=T. Then, as long as there exists a non-fork vertex of degree 3 (namely, with 2 neighbors in F and 1 in T'), remove it and its incident edges from T'. The vertices removed in this step are called *foliage* (small crosses in Fig. 1(b)). All the remaining non-fork vertices have degree 4 (namely 2 in F and 2 in T'); for each of them, remove it and its incident edges from T' and add an edge between the two vertices of T' that were connected to it before its removal. The

vertices removed in this step are *branch* vertices (large crosses in Fig. 1(b)). A vertex $w_x \in F$ is called *free* if so far no vertex of T' has label $\ell(w_x)$. To perform the labeling, we traverse T' bottom-up with respect to a root r that is the vertex of T' adjacent to both w_1 and w_m. Since $[F, T]$ is inner-triangulated, this vertex is unique. During the traversal of T', we maintain the invariant that vertices of T' are incident to only free vertices of F. Initially the invariant is satisfied since all the vertices of F are free. Let a be the fork vertex considered in a step of the traversal of T', and let $w_{a_1}, \ldots, w_{a_k}$ be the vertices of F adjacent to a, with $1 \leq a_1 < \cdots < a_k \leq m$ and $k \geq 3$. By the invariant, $w_{a_1}, \ldots, w_{a_k}$ are free. Choose any vertex w_{a_i} such that $2 \leq i \leq k-1$, and set $\ell(a) = \ell(w_{a_i})$. For example, the red fork vertex in Fig. 1(b) adjacent to w_3, w_4, and w_5 in F gets label $\ell(w_4)$. Since vertices $w_{a_2}, \ldots, w_{a_{k-1}}$ cannot be adjacent to any vertex of T' that is visited after a in the bottom-up traversal, the invariant is maintained at the end of each step. When finally a=r, then $w_{a_1} = w_1$ and $w_{a_k} = w_m$ are both free.

Now we label the non-fork vertices of T based on the labeling of T'. Let b be a non-fork vertex. If b is a branch vertex, then consider the first fork vertex a encountered on a path from b to a leaf of T; set $\ell(b) = \ell(a)$. Otherwise, b is a foliage vertex. In this case, consider the first fork vertex a' encountered on a path from b to the root r of T. Let $v, w \in F$ be the two vertices of F adjacent to b; assume $\ell(v) < \ell(w)$. If $\ell(a') \leq \ell(v)$, then set $\ell(b) = \ell(v)$; if $\ell(a') \geq \ell(w)$, then set $\ell(b) = \ell(w)$; and if $\ell(v) < \ell(a') < \ell(w)$, then set $\ell(b) = \ell(a')$ (the latter case only happens when a' is the root and b is adjacent to w_1 and w_m). Note that the described algorithm ensures that adjacent non-fork vertices have the same label. We perform the labeling procedure for every $T_h \in \mathcal{H}$ and obtain a labeling for $[G, \mathcal{H}]$. We say that the subgraph of $\mathcal{H}$ induced by all the vertices of $\mathcal{H}$ with label i is the *restricted subgraph* H_i of $\mathcal{H}$ for all $i = 1, \ldots, |G|$ (see Fig. 1(b)).

Lemma 1. *Each restricted subgraph H_i of $\mathcal{H}$, $1 \leq i \leq |G|$, is a tree all of whose vertices have degree at most 2, except for one vertex that may have degree 3.*

Proof sketch. First, H_i has at most one fork vertex a, which is hence the only one with degree larger than 2. Further, a is incident to at most one path (to no path, if $a = r$) of branch vertices, namely the one connecting it to its parent fork vertex. Finally, a is incident to at most two (if $a \neq r$) or at most three (if $a = r$) paths of foliage vertices, namely the ones whose vertices are incident to the vertex $w \in F$ such that $\ell(w) = i$. □

3.3 Embedding on the Point Set

We describe an embedding algorithm consisting of three steps (see Fig. 1(b)).
Step a: Let $\omega : G \to \mathbb{N}$ be a weight function with $\omega(v_i)$=$|\{v \in [G, \mathcal{H}] \mid \ell(v) = i\}|$ for every $v_i \in G$. Note that $\sum_{v_i \in G} \omega(v_i) = n$. We categorize each vertex $v_i \in G$ as *sparse* if $1 \leq \omega(v_i) \leq \sqrt{n}$, and *dense* if $\omega(v_i) > \sqrt{n}$. There are at most $\sqrt{n}$ dense vertices.
Step b: We draw the vertices $v_1, \ldots, v_{|G|}$ of G on the $N := n + \sqrt{n}$ points of π in the same order as they appear along the outer face of Γ, in such a way that dense (resp. sparse) vertices are placed on dense (resp. sparse) points. The resulting embedding $\widetilde{\Gamma}$ of G is planar since Γ is planar. The construction of $\widetilde{\Gamma}$ implies the following.

Property 2. Let $Q = \{p_{j_1}, \ldots, p_{j_m}\} \subseteq \pi$, $j_i < j_{i+1}$, be the polygon representing a face of G. Polygon Q contains in its interior all the point sets $S_{j_2}, \ldots, S_{j_{m-1}}$.

Step c: Finally, we consider forest $\mathcal{H} = \{T_1, \ldots, T_k\}$. We describe the embedding algorithm for a single cycle-tree graph $[F, T]$, where $F = w_1, \ldots, w_m$ is a face of G and $T \in \mathcal{H}$ is the tree lying inside F. We show how to embed the restricted subgraph H_i, for each vertex w_x of F with label $\ell(w_x) = i$, on the point set S_j of the point p_j where w_x is placed. We remark that the labeling procedure ensures that $|H_i|+1 = \omega(w_x) \leq |S_j|$; also, by Property 2, point set S_j lies inside the polygon representing F, except for the two points where vertices w_1 and w_m have been placed.

By Lemma 1, H_i has at most one (fork-)vertex a of degree 3, while all other vertices have smaller degree. We place a, if any, on the center point p_j^C of p_j. The at most three paths of non-fork vertices are placed on segments s_j^+, s_j^-, s_j^N starting from p_j^C; namely, the unique path of branch vertices is placed on s_j^N, while the two paths of foliage vertices are placed on s_j^+ or s_j^- based on whether the vertex of G different from w_x they are incident to is w_{x+1} or w_{x-1}, respectively. If $a = r$, then the path of foliage vertices incident to w_1 and w_m is placed on s_j^N.

We show that this results in a planar drawing of T. First, for every two fork vertices $a \in H_p$ and $a' \in H_q$, with $p < q$, all the leaves of the subtree of T rooted at a have smaller label than all the leaves of the subtree of T rooted at a'. Then, for each $w_x \in F$, with $\ell(w_x) = i$, consider the fork vertex $a \in H_i$, which lies on p_j^C. Let P be any path connecting a to a leaf of T and let a^* be the neighbor of a in P. If P contains a fork vertex other than a (Fig. 2(a)), then let a' be the fork vertex in P that is closest to a (possibly a'=a^*) and let p_q^C be the point where a' has been placed. Assume $q < j$, the case $q > j$ is analogous. By definition, the non-fork vertices in the path from a to a' (if any) are branch vertices, and hence lie on s_q^N. Then, Property 1 ensures that the straight-line edge (a, a^*) separates all the point sets S_p with $q < p < j$ from the center of π. Since the vertices on S_p are only connected either to each other or to the vertices on s_j^- and s_q^+, edge (a, a^*) is not involved in any crossing. If P does not contain any fork vertex other than a (Fig. 2(a)), then all the vertices of P other than a are foliage vertices and are placed on a segment s_q^+ or s_q^-, for some q. In particular, if $q < j$, then they are on s_q^-; if $q > j$, then they are on s_q^+; while if $q = j$, then they are either on s_q^+ or on s_q^-. In all the cases, Property 1 ensures that edge (a, a^*) does not cross any edge.

Finally, observe that any path of T containing only non-fork vertices is placed on the same segment of the point set, and hence its edges do not cross. As for the edges connecting vertices in one of these paths to the two leaves of T they are connected to, note that by item (A) of Property 1 the edges between each of these leaves and these vertices appear in the rotation at the leaf in the same order as they appear in the path.

Lemma 2. *There exists a universal point set of size $O(n^{3/2})$ for the class of n-vertex inner-triangulated 2-outerplanar graphs $[G, \mathcal{H}]$ where $\mathcal{H}$ is a forest.*

4 2-Outerplanar Graphs with Forest

In this section we consider 2-outerplanar graphs $[G, \mathcal{H}]$ where $\mathcal{H}$ is a forest. Contrary to the previous section, we do not assume $[G, \mathcal{H}]$ to be inner-triangulated. As observed before, augmenting it might be not possible without introducing multiple edges. The main idea to overcome this problem is to first identify the parts of $[G, \mathcal{H}]$ not allowing for

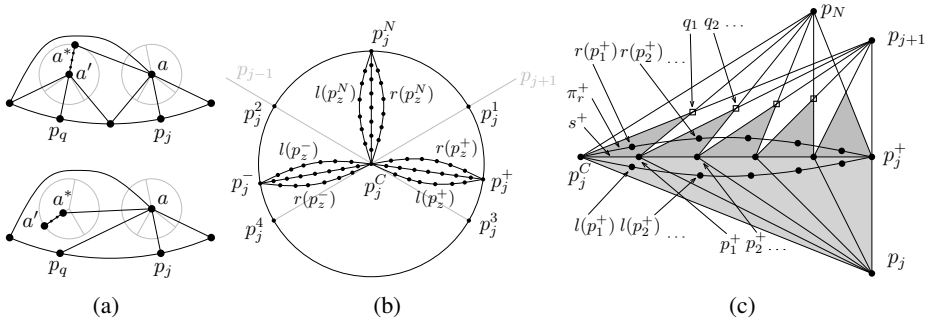


Fig. 2. (a)(top) P contains $a' \neq a$, (a)(bottom) a' is a leaf of T. (b)–(c) Dark-gray triangles are used for construction of petal points $r(p_z^+)$ while light-gray triangles for $l(p_z^+)$.

the augmentation, remove them, and augment the resulting graph with dummy edges to inner-triangulated (Section 4.2); then, apply Lemma 2 to embed the inner-triangulated graph on the point set S; and finally remove the dummy edges and embed the parts of the graph that had been previously removed on the remaining points (Section 4.3). To do so, we first need to extend the point set S with some additional points.

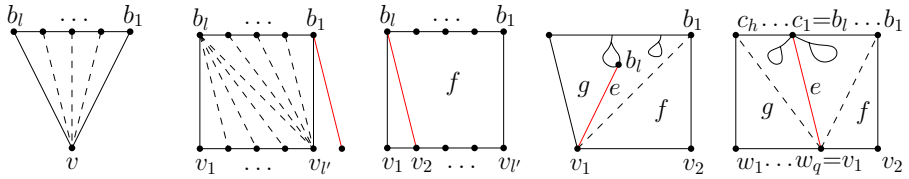
4.1 Extending the Universal Point Set

We construct a point set S^* with $O(n^{3/2})$ points from S by adding *petal points* to segments s_j^+, s_j^N, s_j^- of the point sets S_j, for every j=2, ..., $N-1$. For simplicity of notation, we skip the subscript j whenever possible. We denote by p_z^σ the z-th point on segment s^σ, with $\sigma \in \{+, -, N\}$ and z=1, ..., $\overline{n}$ (where $\overline{n}$=$\sqrt{n}$ or $\overline{n}$=n, depending on whether p_j is sparse or dense), so that p_1^σ is the point following p^C along s^σ and $p_{\overline{n}}^\sigma = p_j^\sigma$. For each point p_z^σ we add two *petal points* $l(p_z^\sigma)$ and $r(p_z^\sigma)$ to S^*.

We first describe the procedure for s^+, see Fig. 2(c). For each z=1, ..., $\overline{n}$, consider the intersection point q_z between segments $s(p_{z-1}^+ p_{j+1})$ and $s(p_z^+ p_N)$, where $p_{z-1}^+ = p_j^C$ when $z = 1$. By construction, all triangles $\triangle p_{z-1}^+ p_z^+ q_z$ have two corners on s^+, have the other corner in the same half-plane delimited by the line through s^+, and do not intersect each other except at common corners. Hence, there exists a convex arc π_r^+ passing through p_j^C and $p_{\overline{n}}^+ = p_j^+$, and intersecting the interior of every triangle. For each $z = 1, \ldots, \overline{n}$, we place the petal point $r(p_z^+)$ on the arc of π_r^+ lying inside triangle $\triangle p_{z-1}^+ p_z^+ q_z$. For the other petal point $l(p_z^+)$ we use the same procedure by considering triangles $\triangle p_{z-1}^+ p_z^+ p_j$ instead of $\triangle p_{z-1}^+ p_z^+ q_z$. Symmetrically we place the petal points for s^-, using points p_{j-1} and p_1 to place $l(p_z^-)$ and point p_j to place $r(p_z^-)$, and for s^N, using points p_{j-1} and p_1 to place $l(p_z^N)$ and points p_{j+1} and p_N to place $r(p_z^N)$.

4.2 Modifying and Labeling the Graph

We now aim at modifying $[G, \mathcal{H}]$ to obtain an inner-triangulated graph that can be embedded on the original point set S (**Part A** and **Part B**); in Section 4.3 we describe



(a) (b) (c) (d) (e)

Fig. 3. (a)–(c): Insertion of triangulation edges in (a) a petal face, (b) a non-protected big face, and (c) a big face protected by vertex b_1. (d)–(e) Illustration of the two cases for removing bad faces. Face g is petal in (d) and big in (e). Dummy edges are dashed, the removed edge e is red.

how to exploit this embedding on S to obtain an embedding of the original graph $[G, \mathcal{H}]$ on the extended point set S^* (**Part C**). We describe the procedure just for a cycle-tree graph $[F, T]$ composed of a face F of G and of the tree T inside it.

Part A: We categorize each face f of $[F, T]$ based on the number of vertices of F and of T that are incident to it. Since T is a tree, f has at least a vertex of F and a vertex of T incident to it. If f contains exactly one vertex of F, then it is a *petal face*. If f contains exactly one vertex of T, then it is a *small face*. Otherwise, it is a *big face*. Let $b_1, \ldots, b_l$ be the occurrences of the vertices of T in a clockwise order walk along the boundary of a big face f. If either b_1 or b_l, say b_1, has more than one adjacent vertex in F (namely one in f and at least one not in f), then f is *protected* by b_1. If f is a big face with exactly two vertices incident to F and is not protected, then f is a *bad face*.

The next lemma gives sufficient conditions to triangulate G without introducing multiple edges; we will later use this lemma to identify the "tree components" of T whose removal allows for a triangulation.

Lemma 3. *Let $[F, T]$ be a biconnected simple cycle-tree graph, such that (1) each vertex of F has degree at most four, and (2) there exists no bad face in $[F, T]$. It is possible to augment $[F, T]$ to an inner-triangulated simple cycle-tree graph.*

Proof sketch. Each petal (small, respectively) face f can be triangulated by adding vertices between the only vertex of F (of T) incident to f and all the other vertices of f. Multiple edges are not created since $[F, T]$ is biconnected and there exists no two petal faces incident to the same vertex v of F, as v has degree at most 4; see Fig. 3(a).

Consider a big face f, with vertex occurrences $v_1, ..., v_{l'}, b_1, ..., b_l$ (with $l, l' > 1$), where $v_1, ..., v_{l'} \in F$ and $b_1, ..., b_l \in T$. If f is protected by a vertex, say b_1, then it is triangulated by adding an edge between b_l and every vertex of F, and an edge between $v_{l'}$ and every vertex of T; see Fig. 3(b). The absence of multiple edges is due to the edge connecting b_1 to a vertex of F not incident to f, which implies that $v_{l'}$ is not connected to any vertex of T incident to f other than b_1. Finally, if f is not protected by any vertex, we make it protected by adding an edge (b_l, v_2) and apply the previous case; see Fig. 3(c). Since f is not a bad face, we have $l' > 2$, and hence v_2 is not connected to any vertex of T, which implies that (b_l, v_2) is not a multiple edge. □

We now describe a procedure to transform cycle-tree graph $[F, T]$ into another one $[F, T'']$ that is biconnected and satisfies the conditions of Lemma 3. We do this in two

steps: first, we remove some edges connecting a vertex of F and a vertex of T to transform $[F, T]$ into a cycle-tree graph $[F, T'{=}T]$ that is not biconnected but that satisfies the two conditions; then, we remove the "tree components" of T' that are not connected to vertices of F in order to obtain a cycle-tree graph $[F, T'' \subseteq T']$ that is also biconnected.

To satisfy condition (1) of Lemma 3, we merge all the petal faces incident to the same vertex of F into a single one by repeatedly removing an edge shared by two adjacent petal faces. We refer to these removed edges as *petal edges*, denoted by E_P.

To satisfy condition (2) of Lemma 3, we consider each bad face $f = v_1, v_2, b_1, \ldots, b_l$, where $v_1, v_2 \in F$ and $b_1, \ldots, b_l \in T$. Let g be the face incident to v_1 sharing edge $e = (v_1, b_l)$ with f. We remove e, hence merging f and g into a single face f', that we split again by adding dummy edges, based on the type of face g, in such a way that no new bad face is created. Since f is a bad face, it is not protected by b_l, and hence g is not a small face. If g is a petal face, then f' is still a big face with two vertices of F incident to it, namely v_1 and v_2; see Fig. 3(d). We add edge (v_1, b_1), splitting f' into a petal face $v_1, b_1, \ldots, b_l$ and a triangular face v_1, v_2, b_1. If g is a big face, then f' is a big face; see Fig. 3(e). Let $g = w_1, \ldots, w_q, c_1, \ldots, c_h$, where $w_1, \ldots, w_q \in F$, with $w_q = v_1$, and $c_1, \ldots, c_h \in T$, with $c_1 = b_l$. We add two dummy edges (v_1, c_h) and (v_1, b_1), splitting f' into a small face $w_1, \ldots, w_q, c_h$, a petal face $v_1, b_1, \ldots, b_l = c_1, \ldots, c_h$, and a triangular face v_1, v_2, b_1. The edges removed in this step are *big face edges*, denoted by E_B, and the added edges are *triangulation edges*.

In order to make $[F, T']$ biconnected, note that $[F, T']$ consists of a biconnected component which contains F, called *block-component*, and a set $\mathcal{T}_B$ of subtrees of T', called *tree components*, each sharing a cut-vertex with the block component. We remove the tree components $\mathcal{T}_B$ from $[F, T']$ and obtain an instance $[F, T'' \subseteq T']$, that is actually the block component of $[F, T']$. Since the removal of $\mathcal{T}_B$ does not change the degree of the vertices of F and does not create any bad face, $[F, T'']$ is indeed a biconnected instance satisfying the two conditions of Lemma 3. Thus, by adding further *triangulation edges* we augment it to an inner-triangulated instance $[F, T^{\Delta} = T'']$.

Lemma 4. *Let $e{=}(b, v)$ be an edge of $E_P \cup E_B$, where $b \in T$ and $v \in F$. Then, either e is a triangulation edge in $[F, T^{\Delta}]$ or b belongs to a tree component T_c of $\mathcal{T}_B$ sharing a cut-vertex c with $[F, T'']$. In the latter case, (v, c) is a triangulation edge in $[F, T^{\Delta}]$.*

Lemma 5. *Let $T_c \in \mathcal{T}_B$ be a tree component such that there exists at least an edge $(b, v) \in E_P \cup E_B$, with $b \in T_c$ and $v \in F$. Then, for each edge in $E_P \cup E_B$ with an endvertex belonging to T_c, the other endvertex is v.*

Performing the above operations for every cycle-tree graph $[F, T]$ yields an inner-triangulated 2-outerplanar graph $[G, \mathcal{H}^{\Delta}]$ as outcome of **Part A**. We then label $[G, \mathcal{H}^{\Delta}]$ with the algorithm from Section 3.2 and describe next how to extend this labeling to $\mathcal{T}_B$.
Part B: We consider the tree components $T_c \in \mathcal{T}_B$ for each face F of G; let $[F, T^{\Delta}]$ be the corresponding inner-triangulated cycle-tree graph. We label the vertices of T_c and simultaneously augment $[F, T^{\Delta}]$ with dummy vertices and edges, so that $[F, T^{\Delta}]$ remains inner-triangulated (and hence can be embedded, by Lemma 2) and the vertices of T_c can be later placed on the petal points of the points where dummy vertices are placed. The face of $[F, T'']$ to which T_c belongs might have been split into several faces

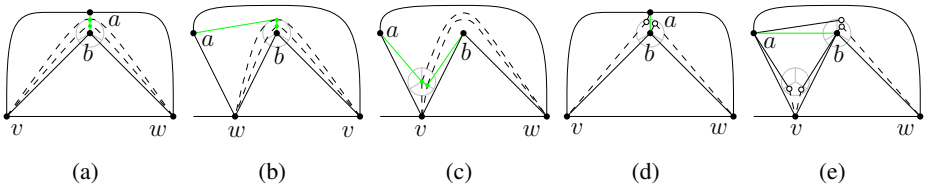


Fig. 4. (a)–(c) Inserting dummy vertices for a tree-component in a face (a, b, v) with $v \in F$ and $a, b \in T^{\Delta}$, when (a) $\ell(a) = \ell(b)$, (b) $\ell(a) \neq \ell(b)$ and $\ell(w) < \ell(v)$, and (c) $\ell(a) \neq \ell(b)$ and $\ell(w) > \ell(v)$. (d)–(e) Moving dummy vertices to petal points if $\ell(a) = \ell(b)$ and if $\ell(a) \neq \ell(b)$.

of $[F, T^{\Delta}]$ by triangulation edges. We assign T_c to any of such faces f that is incident to the root c of T_c. Then, we label T_c based on the type of f; we distinguish two cases.

Suppose f is a triangular face (c, v, w) with $v, w \in F$ and $c \in T^{\Delta}$; assume $\ell(v) < \ell(w)$. We create a path P_c containing $|T_c| - 1$ dummy vertices and append P_c at c. Then, we connect every dummy vertex of P_c with both v and w. If $\ell(c) \leq \ell(v)$, then we label the vertices of P_c with $\ell(P_c) = \ell(v)$. If $\ell(c) \geq \ell(w)$, then we label $\ell(P_c) = \ell(w)$.

Suppose f is a triangular face (a, b, v) with $v \in F$ and $a, b \in T^{\Delta}$, refer to Fig. 4; assume $\ell(a) \leq \ell(b)$. Replace edge (a, b) with a path P_c between a and b with $|T_c| - 1$ internal dummy vertices, and connect each of them to v and to w, where w is the other vertex of F adjacent to both a and b. For each dummy vertex x of P_c, we assign $\ell(x) = \ell(a)$ if $\ell(v) \leq \ell(a)$; we assign $\ell(x) = \ell(b)$ if $\ell(v) \geq \ell(b)$; and we assign $\ell(x) = \ell(v)$ if $\ell(a) < \ell(v) < \ell(b)$. The existence of edge $(a, b) \in T^{\Delta}$ implies that either a is the parent of b in T^{Δ} or vice versa. Suppose the former, the other case is analogous. Then, v and w are the extremal neighbors of b in F, and thus either $\ell(v) \leq \ell(b) \leq \ell(w)$ or $\ell(w) \leq \ell(b) \leq \ell(v)$. Also, if $\ell(a) \neq \ell(b)$, then $\ell(a)$ does not lie strictly between $\ell(v)$ and $\ell(w)$. In fact, this can only happen if $\ell(b)$ strictly lies between $\ell(v)$ and $\ell(w)$, and $\ell(a) = \ell(b)$ (which happens only if a is a non-fork vertex). Since $\ell(a) \leq \ell(b)$, by assumption, this implies that $\ell(a) \leq \ell(v), \ell(w)$. The two observations before can be combined to conclude that, if $\ell(a) = \ell(b)$, then all the tree components lying inside faces (a, b, v) and (a, b, w) have the same label as a and b (Fig. 4(a)). Otherwise, either the tree components inside (a, b, v) have label $\ell(b)$ and those inside (a, b, w) have label $\ell(w)$ (Fig. 4(b)), or the tree components inside (a, b, v) have label $\ell(v)$ and those inside (a, b, w) have label $\ell(b)$ (Fig. 4(c)). All added edges are again *triangulation edges*.

We apply **Part B** to every cycle-tree graph of $[G, \mathcal{H}^{\Delta}]$, hence creating an inner-triangulated 2-outerplanar graph $[G, \mathcal{H}^{\mathcal{A}}]$ where $\mathcal{H}^{\mathcal{A}}$ is a forest. Since all the dummy vertices of P_c are connected to two vertices $v, w \in F$, they become non-fork vertices. Note that the labeling of the dummy vertices coincides with the one obtained by the algorithm in Section 3.2, except for the case when f is a triangular face (a, b, v) with $v \in F$ and $a, b \in T^{\Delta}$, and $\ell(a) < \ell(v) < \ell(b)$. In this case the algorithm would have labeled either $\ell(P_c) = \ell(a)$ or $\ell(P_c) = \ell(b)$, depending on whether b is the parent of a or vice versa. However, since $\ell(a) < \ell(v) < \ell(b)$ holds in $[F, T^{\Delta}]$, and since (a, b, v) is a triangular face of $[F, T^{\Delta}]$, no vertex of $[F, T^{\Delta}]$ different from v has the same label as v. Hence, graph H_i, for each i, is a tree with at most one vertex of degree 3. We thus apply Lemma 2 to obtain a planar embedding $\Gamma^{\mathcal{A}}$ of $[G, \mathcal{H}^{\mathcal{A}}]$ on S.

4.3 Transformation of the Embedding

We remove the all the triangulation edges added in the construction, and then restore each tree component T_c, which is represented by path P_c. Since the vertices of P_c are non-fork vertices and have the same label i, by construction, they are placed on the same segment $s \in \{s^+, s^N, s^-\}$ of S_j, where p_j is the point vertex v_i is placed on.

We remove all the internal edges of P_c and move each vertex x of P_c from the point p of s it lies on to one of the corresponding petal points, either $l(p)$ or $r(p)$, as follows. Let v be a vertex of G connected to a vertex of T_c by an edge in $E_P \cup E_B$, if any; recall that, by Lemma 5, all the edges of $E_P \cup E_B$ connecting T_c to G are incident to v. If $\ell(x) < \ell(v)$, then move x to $r(p)$; tree components connected to w in Fig. 4(d) and 4(e). If $\ell(x) > \ell(v)$, then move x to $l(p)$; tree component connected to v in Fig. 4(e). Otherwise, $\ell(x) = \ell(v)$; in this case $s \neq s^N$, by construction, and hence we have to distinguish the following two cases: If $s = s^+$, then move x to $l(p)$, otherwise move x to $r(p)$ (tree components attached to a and b, respectively, and connected to v in Fig. 4(e)). If no vertex $v \in G$ is connected to T_c, then move x to $r(p)$ if $\ell(c) < \ell(x)$ (tree component attached to a in Fig. 4(e)), and to $l(p)$ otherwise.

We prove that this operations maintain planarity. The internal edges of T_c do not cross since the petal points, together with the point where c lies, form a convex point set, on which it is possible to construct a planar embedding of every tree [4]. As for the edges connecting vertices of T_c to v, by Lemma 4, v has visibility to the root c of T_c, since (v, c) is a triangulation edge; by Property 1, this visibility from v extends to all the segment s where P_c had been placed on; and by the construction of S^*, to all the corresponding petal points. The proof for the edges (a, b) that had been subdivided when merging tree component T_c (green edges in Fig. 4(d) and 4(e)) is in [2].

Claim 1 *Reinserting every edge (a, b) such that there existed a path P_c between a and b does not introduce any crossing.*

To complete the transformation it remains to insert the edges of $E_P \cup E_B$ which were not inserted in the previous step. Since by Lemma 4 all of these edges were also triangulation edges, their insertion does not produce any crossing.

Lemma 6. *There exists a universal point set of size $O(n^{3/2})$ for the class of n-vertex 2-outerplanar graphs $[G, \mathcal{H}]$ where $\mathcal{H}$ is a forest.*

5 General 2-Outerplanar Graphs

In this section we give a high-level idea of how to extend the result of Lemma 6 to any arbitrary 2-outerplanar graph $[G, \mathcal{H}]$. The complete description can be found in [2].

The idea is to convert every graph $G_h \in \mathcal{H}$ into a tree T_h; embed the resulting graph on S^*; and finally revert the conversion from each T_h to G_h. Each tree T_h is created by substituting each biconnected block B of G_h by a star, centered at a dummy vertex and with a leaf for each vertex of B, where leaves shared by more stars are identified. This results in a 2-outerplanar graph whose inner level is a forest.

The embedding of this graph on S^* is performed similarly as in Lemma 6, with some slight modifications to the labeling algorithm, especially for the vertices of T_h

corresponding to cut-vertices of G_h, and to the procedure for merging the tree components. These modifications allow us to ensure that the vertices of each block of G_h lie on a convex portion of S^*, where they can thus be drawn without crossings [5, 10].

We finally reduce the size of S^* to $O(n \log n)$ by using the super-pattern sequence ξ from [3], which is a sequence of integers ξ_j, with $\sum_{j=1,\ldots,n} \xi_j = O(n \log n)$. Sequence ξ majorizes every sequence of integers that sum up to n. We hence assign the size of each point set S_j based on this sequence, instead of using dense or sparse point sets.

Theorem 1. *There exists a universal point set of size $O(n \log n)$ for the class of n-vertex 2-outerplanar graphs.*

6 Conclusions

We provided a universal point set of size $O(n \log n)$ for 2-outerplanar graphs. A natural question is whether our techniques can be extended to other meaningful classes of planar graphs, such as 3-outerplanar graphs. We also find interesting the question about the required area of universal point sets. In fact, while the integer grid is a universal point set for planar graphs with $O(n^2)$ points and $O(n^2)$ area, all known point sets of smaller size, even for subclasses of planar graphs, require a larger area. We thus ask whether universal point sets of subquadratic size require polynomial or exponential area.

References

1. P. Angelini, G. D. Battista, M. Kaufmann, T. Mchedlidze, V. Roselli, and C. Squarcella. Small point sets for simply-nested planar graphs. In M. van Kreveld and B. Speckmann, editors, *Graph Drawing*, volume 7034 of *LNCS*, pages 75–85. Springer, 2012.
2. P. Angelini, T. Bruckdorfer, M. Kaufmann, and T. Mchedlidze. A universal point set for 2-outerplanar graphs. *CoRR*, abs/1508.05784, 2015.
3. M. J. Bannister, Z. Cheng, W. E. Devanny, and D. Eppstein. Superpatterns and universal point sets. *J. Graph Algorithms Appl.*, 18(2):177–209, 2014.
4. C. Binucci, E. Di Giacomo, W. Didimo, A. Estrella-Balderrama, F. Frati, S. Kobourov, and G. Liotta. Upward straight-line embeddings of directed graphs into point sets. *CGTA*, 43:219–232, 2010.
5. P. Bose. On embedding an outer-planar graph in a point set. *CGTA*, 23(3):303–312, 2002.
6. S. Cabello. Planar embeddability of the vertices of a graph using a fixed point set is NP-hard. *J. Graph Algorithms Appl.*, 10(2):353–366, 2006.
7. H. de Fraysseix, J. Pach, and R. Pollack. Small sets supporting fáry embeddings of planar graphs. In J. Simon, editor, *STOC '88*, pages 426–433. ACM, 1988.
8. H. Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10:41–51, 1990.
9. R. Fulek and C. D. Tóth. Universal point sets for planar three-trees. *J. Discrete Algorithms*, 30:101–112, 2015.
10. P. Gritzmann, B. M. J. Pach, and R. Pollack. Embedding a planar triangulation with vertices at specified positions. *American Mathematical Monthly*, 98:165–166, 1991.
11. M. Kurowski. A 1.235 lower bound on the number of points needed to draw all n-vertex planar graphs. *Information Processing Letters*, 92(2):95–98, 2004.
12. W. Schnyder. Embedding planar graphs on the grid. In D. S. Johnson, editor, *SODA '90*, pages 138–148. SIAM, 1990.