Algorithmica (2016) 74:1293–1320 (Preprint)
DOI 10.1007/s00453-015-0002-1

Outer 1-Planar Graphs

Christopher Auer · Christian Bachmaier · Franz J. Brandenburg · Andreas Gleißner · Kathrin Hanauer · Daniel Neuwirth · Josef Reislhuber

Received: 14 November 2013 / Accepted: 13 April 2015 / Published online: 22 April 2015

Abstract A graph is outer 1-planar (*o1p*) if it can be drawn in the plane such that all vertices are in the outer face and each edge is crossed at most once. *o1p* graphs generalize outerplanar graphs, which can be recognized in linear time, and specialize 1-planar graphs, whose recognition is NP-hard.

We explore *o1p* graphs. Our first main result is a linear-time algorithm that takes a graph as input and returns a positive or a negative witness for *o1p*. If a graph G is *o1p*, then the algorithm computes an embedding and can augment G to a maximal *o1p* graph. Otherwise, G includes one of six minors, which is detected by the recognition algorithm.

Secondly, we establish structural properties of *o1p* graphs. *o1p* graphs are planar and are subgraphs of planar graphs with a Hamiltonian cycle. They are neither closed under edge contraction nor under subdivision. Several important graph parameters, such as treewidth, colorability, stack number, and queue number, increase by one from outerplanar to *o1p* graphs. Every *o1p* graph of size n has at most $\frac{5}{2}n - 4$ edges and there are maximal *o1p* graphs with $\frac{11}{5}n - \frac{18}{5}$ edges, and these bounds are tight.

Finally, every *o1p* graph has a straight-line grid drawing in $\mathcal{O}(n^2)$ area with all vertices in the outer face, a planar visibility representation in $\mathcal{O}(n \log n)$ area, and a 3D straight-line drawing in linear volume, and these drawings can be constructed in linear time.

Keywords planar and outerplanar graphs · 1-planarity · embeddings and drawings · graph parameters · density

1 Introduction

Planar graphs have been intensively studied in graph theory and graph drawing. Outerplanar graphs are an important subfamily of planar graphs. Here, all vertices are in the outer face

This work was supported in part by the Deutsche Forschungsgemeinschaft (DFG) grant Br835/18-1.

University of Passau
94030 Passau
Tel.: +49/851/509-3031
Fax: +49/851/509-3032
E-mail: {auerc,bachmaier,brandenb,gleissner,hanauer,neuwirth,reislhuber}@fim.uni-passau.de

and edges do not cross. This implies several structural properties. Every outerplanar graph has at least two vertices of degree two, which is utilized in a linear-time recognition algorithm [36]. The complete graph K_4 and the complete bipartite graph $K_{2,3}$ are not outerplanar; in fact, they are the forbidden minors. The weak dual of a maximal outerplanar graph is a binary tree and outerplanar graphs have at most $2n-3$ edges. They are 3-colorable and have treewidth at most two, stack number one [8] and queue number at most two [30], and these bounds are tight. Every outerplanar graph admits a planar straight-line drawing within an area of $\mathcal{O}(dn\log n)$ [26] for graphs of degree d or $\mathcal{O}(n^{1.48})$ [17]. Additionally, there is a visibility representation in $\mathcal{O}(n\log n)$ area [9].

There were several approaches to generalize planarity to graphs that are "almost" planar in some sense. Such attempts are important as many graphs are not planar, and it is desirable to transfer properties beyond planarity. One generalization is 1-planarity, which was introduced by Ringel [38] in an approach to color a planar graph and its dual. A graph is 1-planar if it can be drawn in the plane such that each edge is crossed at most once. 1-planar graphs have recently obtained much attention, see also [1, 12, 14, 15, 22, 23, 32, 34].

The combination of 1-planar and outerplanar leads to *o1p* graphs, which are graphs with an embedding in the plane with all vertices in the outer face and at most one crossing per edge. They were introduced by Eggleton [24], who called them outerplanar graphs with edge crossing number one. He showed that edges of maximal *o1p* graphs do not cross in the outer face and that each face is incident to at most one crossing, from which he concluded that every *o1p* graph has an *o1p* drawing with straight-line edges and convex (inner) faces. Thomassen [42] generalized Eggleton's result and characterized the class of 1-planar graphs which admit straight-line drawings by the exclusion of so-called B- and W-configurations in embeddings. These configurations were rediscovered by Hong et al. [32], who also provide a linear-time drawing algorithm that starts from a given embedding.

From the algorithmic perspective there is a big step from zero to some crossings. It is well-known that planar graphs can be recognized in linear time, and there are linear-time algorithms to construct an embedding, maximal augmentations, and drawings, e. g., straight-line drawings and visibility representations in quadratic area. On the contrary, dealing with crossings generally leads to NP-hard problems. It is NP-hard to recognize 1-planar graphs [34], even if the graph is given with a rotation system, which determines the cyclic ordering of the edges at each vertex [6]. 1-planarity remains NP-hard even for bounded treewidth [7]. There is no efficient algorithm to compute the crossing number of a graph [31] or to compute the number of crossings induced by the insertion of an edge into a planar graph [14]. However, there is a linear-time recognition algorithm for maximal 1-planar graphs if the rotation system is given [22].

In this paper we thoroughly investigate *o1p* graphs. One major result is a linear-time recognition algorithm for *o1p* graphs. In the context of 1-planarity, it is the first efficient algorithm that returns a positive or negative witness in terms of either an *o1p* embedding or one of six minors. As such it resembles advanced planarity testing algorithms, which either return a planar embedding or detect a minor [44]. In contrast, *o1p* graphs are neither closed under edge contraction nor under subdivision. Hence, there is no characterization of *o1p* graph by forbidding some minors as it is known for example for planar graphs. Our algorithm[1] works directly on SPQR-trees, analyzes the structure of its nodes, and determines whether an edge is plane or crossed in every embedding, or whether this depends on the concrete embedding. Independently, Hong et al. [33] obtained a linear-time testing algorithm, which returns an *o1p* embedding in the positive case.

[1] A short version of the algorithm appeared in [5].

If the graph is *o1p*, it can be augmented to a maximal *o1p* graph. To a large extent, this is already done by our recognition algorithm. A graph is *maximal* for a class of graphs if adding a new edge violates its defining property. Maximal graphs often provide deep insights into graph properties. First, we derive that *o1p* graphs are planar. In fact, they are subgraphs of planar graphs with a Hamiltonian cycle, which are the 2-stack graphs [8]. This is due to the fact that *o1p* graphs have an underlying tree structure, which finds expression in a simplified planar dual graph and results in treewidth at most three. The simplified dual of a maximal *o1p* graph is a ternary tree, whose nodes correspond to K_3s and K_4s. From these trees we obtain that every *o1p* graph of size n has at most $\frac{5}{2}n-4$ edges and that there are sparse maximal *o1p* graphs with $\frac{11}{5}n-\frac{18}{5}$ edges. The upper bound is $\frac{n}{2}-1$ above the respective value for outerplanar graphs. Both upper and lower bounds are tight. Hence, there is a fixed interval for the density of maximal *o1p* graphs. This parallels results for maximal 1-planar graphs [12], where it was shown that there are sparse maximal 1-planar graphs with only $\frac{45}{17}n+\mathscr{O}(1)$ edges and that every maximal 1-planar graph has at least $\frac{21}{10}n+\mathscr{O}(1)$ edges. The upper bound of $4n-8$ edges was proved independently by several authors [10, 25, 37].

Moreover, important graph parameters, such as treewidth, chromatic number (coloring), stack number, and queue number increase by one from outerplanar to *o1p* and are 3, 4, 2, and 3, respectively. For a particular graph, these numbers (except for the queue number) can be computed efficiently for *o1p*, which contrasts with the situation for planar graphs, where this is open for treewidth, whereas chromatic number [28], stack number [43], and queue number [30] remain NP-hard.

Finally, we investigate drawings. Every *o1p* graph has a straight-line grid drawing in quadratic area, since *o1p* graphs are planar. Dehkordi and Eades [15] proved that every *o1p* drawing can be transformed into a right angle crossing drawing, but at the expense of exponential area. Here, all edges are straight lines and edges cross at a right angle. We show that *o1p* graphs have a straight-line grid drawing in $\mathscr{O}(n^2)$ area, where all vertices are in the outer face. Furthermore, they have a planar visibility representation in $\mathscr{O}(n\log n)$ area and a 3D straight-line drawing in linear volume. These drawings can be computed in linear time from an input graph.

2 Preliminaries

We consider simple, undirected graphs $G=(V,E)$ with n vertices and m edges. Two vertices are a *separation pair* if their removal disconnects the graph. A *drawing* of a graph is a mapping of G into the plane such that the vertices are mapped to distinct points and each edge is a Jordan arc between its endpoints. A drawing is *planar* if the (Jordan arcs of the) edges do not cross and is *1-planar* if each edge is crossed at most once. Accordingly, a graph is planar (1-planar) if it has a planar (1-planar) drawing. Crossings of edges with the same endpoint, i. e., *incident* edges, are excluded since their local order can be swapped at their common vertex in order to avoid such crossings. Similarly, self-intersections of edges can always be avoided and are excluded. A planar drawing of a graph partitions the plane into *faces*. A face is specified by a cyclic sequence of edges that forms its boundary. The set of all faces forms the *embedding* of the graph. In 1-planar drawings, every crossing divides an edge into two *edge segments*. An uncrossed edge consists of one segment. Therefore, a face of a *1-planar embedding* is specified by a cyclic list of edge segments. Replacing every pair of crossing edges by a new vertex of degree four yields the *planarization* of G with respect to this embedding.

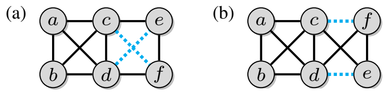


Fig. 1: (a) Chain of two K_4s. (b) After flipping the right K_4 (vertices e, f).

A graph G is *outerplanar* if it has a planar drawing with all vertices in one distinguished face. This face is referred to as the *outer face* and corresponds to the unbounded, external face in a drawing in the plane. G is *maximal* outerplanar if no further edge can be added without violating outerplanarity. Then, the edges in the outer face form a Hamiltonian cycle. A graph G is *outer 1-planar*, *o1p* for short, if it has a drawing with all vertices in the outer face and such that each edge is crossed at most once. G is *maximal o1p* if the addition of any edge violates outer 1-planarity, and *plane-maximal o1p* if no edge can be added without inducing a crossing or violating outerplanarity. In an *o1p* embedding, an edge is either *crossing* or *plane* (*non-crossing*). We say that it is *inner*, if none of its segments is part of the boundary of the outer face. Analogously, an edge is *outer*, if it is entirely part of this boundary. Observe that a crossed edge cannot be outer. If the embedding is maximal, then no crossing is on the outer face [24, 40] and hence we can classify every edge as *outer* or *inner*.

Maximal outerplanar graphs have a unique embedding up to inversion. This does no longer hold for maximal *o1p* graphs. Consider a graph with 6 vertices and 11 edges consisting of two K_4s as depicted in Fig. 1(a). If the left K_4 is fixed, the right can be flipped (Fig. 1(b)), which also changes the pair of crossing edges. However, we show that there is a maximal *o1p* embedding if and only if all *o1p* embeddings are maximal.

2.1 SPQR-trees

In order to gain more insight into the structure of an *o1p* graph G, we consider its *SPQR-tree* $\mathscr{T}$. SPQR-trees were introduced by Di Battista and Tamassia [19] and provide a description of how a biconnected graph is composed of triconnected components, series and parallel compositions. In the following, we give a short introduction into this data structure and refer to [19] and [29] for a more details.

In the definition we adopt here, the SPQR-tree is unrooted. An example is provided in Fig. 3, which shows a graph in Fig. 3(a) along with its SPQR-tree in Fig. 3(b). The SPQR-tree is built upon separation pairs. To demonstrate this, consider the following splitting operation: Let $\{u, v\}$ be a separation pair of G and $G_1, \dots, G_k$ be the connected components obtained after the removal of u and v from G. Partition the set of connected components into two and rejoin each partition such that we obtain two subgraphs G' and G'' of G which both contain at least two edges. Finally, insert a new, so-called *virtual edge* $\{u, v\}$ into G' and into G'', even if this creates a multi-edge. In G', the inserted edge $\{u, v\}$ is meant to represent the subgraph of G that corresponds to G'' and vice versa for the inserted edge $\{u, v\}$ in G''. This mutual relationship is expressed by linking the inserted edges $\{u, v\}$ in G' and G''.

The reverse operation to splitting is a *2-clique-sum* at the linked edges: Given two graphs G' and G'' whose edges $\{u, v\}$ are linked, the 2-clique-sum $G' \oplus G''$ is obtained by merging the vertices u, respectively v, in G' and G'' and removing the linked edges $\{u, v\}$.

By applying the splitting operation recursively to G' and G'', we finally obtain components that are either a cycle of length 3, consist of two vertices and three parallel edges, or are triconnected. We label the components that fall into the first category with S for series composition, those in the second with P for parallel composition and the latter with R for rigid. This decomposition of G depends on the order of splits and is not unique yet. It becomes unique by forming again the 2-clique-sum of two components that have linked edges if both are labeled S or both are labeled P, and this yields the SPQR-tree $\mathscr{T}$.

The SPQR-tree $\mathscr{T}$ consists of a node μ for each component, which is referred to as the node's *skeleton* $\mathrm{skel}(\mu)$. Subsequently, each node represents either a series composition (S), a parallel composition (P), or a triconnected component (R). By construction, every skeleton is homeomorphic to a subgraph of G. If two components had a linked edge $\{u,v\}$ and the components are the skeletons of nodes μ and ν, then $\mathscr{T}$ contains an edge between μ and ν and μ and ν are called *adjacent* in $\mathscr{T}$. We also keep the term virtual edge for the respective edges in skeletons. Let e denote the virtual edge $\{u,v\}$ in $\mathrm{skel}(\mu)$ and e' denote the virtual edge $\{u,v\}$ in $\mathrm{skel}(\nu)$. We say that ν is the *refining* node $\mathrm{refn}(e)$ of e and, symmetrically, μ is the refining node $\mathrm{refn}(e')$ of e'. The whole subgraph of G that is represented by a virtual edge e is called the *expansion graph*[2] $\mathrm{expg}(e)$ of e. As a result of the last step in the construction of the SPQR-tree, neither two S- nor two P-nodes are adjacent in $\mathscr{T}$, the skeleton of every S-node is a cycle of length at least three, and the skeleton of every P-node consists of exactly two vertices and at least three parallel edges.

In the definition given in [19], an SPQR-tree additionally has Q-nodes, which represent one edge of G at a time. Consequently, there every skeleton of an S-, P-, or R-node has only virtual edges. For simplification, we omit Q-nodes and have both virtual and *non-virtual* edges in the skeletons of the nodes.

An interesting feature of SPQR-trees is their ability to represent all planar embeddings of a planar graph via the embeddings of the skeletons [19, 35]. To this end, choose a planar embedding of every skeleton of the SPQR-tree and re-build the graph along with a planar embedding using 2-clique-sums. Let G' and G'' be two graphs with plane embeddings $\mathscr{E}'$ and $\mathscr{E}''$, respectively, and let the edges e' in G' and e'' in G'' be linked. Build the 2-clique-sum as described above and merge $\mathscr{E}'$ and $\mathscr{E}''$ such that $\mathscr{E}''$ with e'' removed takes the position of e' in $\mathscr{E}'$ and vice versa to obtain a planar embedding for the combined graph. Consequently, testing planarity of a graph can be reduced to testing planarity of the skeletons of the R-nodes, which is known as testing the triconnected components.

3 Recognition

There are linear-time algorithms for the recognition of (maximal) outerplanar graphs that use the fact that there are at least two vertices of degree two. A single K_4 implies that this property no longer holds for *o1p* graphs. In contrast, the recognition of 1-planar graphs is NP-hard [34], even if the graphs are given with a rotation system [6].

3.1 Finding an *o1p* Embedding

Theorem 1 *There is a linear-time algorithm to test whether a graph G is o1p and, if so, returns an o1p embedding.*

[2] In the rooted version of SPQR-trees the expansion graph $\mathrm{expg}(e)$ corresponds to the pertinent graph of $\mathrm{refn}(e)$.

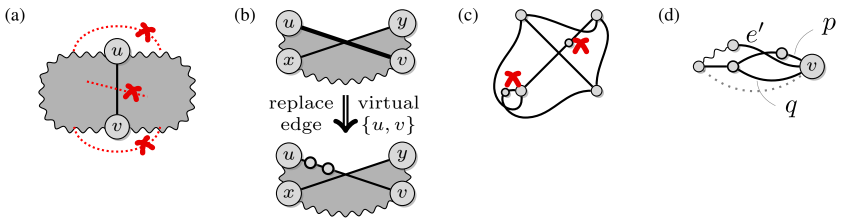


Fig. 2: (a) Proposition 2, (b) Proposition 3, (c) Corollary 2, (d) a “partial” crossing.

For the proof we first establish some necessary conditions for a graph G to have an $o1p$ embedding. At the same time, we implement a linear-time algorithm that checks these conditions and, if positive, constructs an $o1p$ embedding of G. In a second step, we show that the conditions are sufficient.

We start with two observations regarding $o1p$ embeddings. It is well-known that every crossing induces a K_4 in maximal 1-planar embeddings. This holds in an even tighter form for $o1p$ embeddings:

Proposition 1 **([15])** *Let $\{a,b\}$ and $\{c,d\}$ be a pair of crossing edges in an $o1p$ embedding of a maximal $o1p$ graph. Then the vertices a, b, c, and d form a K_4 and the edges $\{a,b\}$, $\{b,c\}$, $\{c,d\}$, and $\{d,a\}$ are plane.*

Consider a plane, inner edge $\{u,v\}$ in an $o1p$ embedding of a graph G. Then $\{u,v\}$ partitions the embedding and the deletion of u and v disconnects G (cf. Fig. 2(a)).

Proposition 2 *For every plane, inner edge $\{u,v\}$, the vertices u, v are a separation pair.*

As an $o1p$ embedding requires every vertex to lie in the outer face, it suffices to test each biconnected component separately and combine the individual embeddings at the cut vertices afterwards. Consequently, we assume biconnectivity for the remainder of this section.

Let $\mathscr{T}$ be the SPQR-tree of an $o1p$ graph G. As mentioned in Sect. 2.1, all planar embeddings of G can be obtained by considering all combinations of all planar embeddings of the skeletons of $\mathscr{T}$. We now want to achieve an analogon for $o1p$ embeddings, i. e., instead of considering the possible $o1p$ embeddings of G all at once, we define *$o1p$ embeddings of skeletons* such that an $o1p$ embedding for G can be obtained in a similar manner as in the planar case. To decide whether a graph is $o1p$, it suffices to show that there exists a $o1p$ embedding and we need not show that all $o1p$ embeddings of G can be represented by a combination of $o1p$ embeddings of the skeletons of $\mathscr{T}$.

Mainly because of virtual edges in conjunction with crossings, we have to distinguish strictly between $o1p$ embeddings of graphs and skeletons. A virtual edge e is embedded plane in a skeleton's embedding if and only if no edge of $\text{expg}(e)$ crosses an edge that is not contained in $\text{expg}(e)$. A virtual edge e crosses another virtual edge e' in a skeleton's embedding if and only if at least one edge of $\text{expg}(e)$ crosses an edge of $\text{expg}(e')$. Likewise, a virtual edge e crosses a non-virtual edge e' in a skeleton's embedding if and only if an edge of $\text{expg}(e)$ crosses e'. One condition that we may adopt directly for $o1p$ embeddings of skeletons is that there must be a face (the outer face) such that all vertices lie on its boundary.

Lemma 1 *The skeleton of every R-node is a K_4.*

Proof Recall that outerplanar graphs are subgraphs of series-parallel graphs. Hence, the SPQR-tree of an outerplanar graph has no R-nodes. Let μ be an R-node in $\mathscr{T}$. Then $\text{skel}(\mu)$

must be embedded such that at least two edges cross, e. g., edges $\{a,b\}$ and $\{c,d\}$. By Proposition 1, the vertices a, b, c, and d either already form a K_4 or the missing edges can be inserted.
There must be an embedding of skel(μ) such that all vertices are on the boundary of the same face. Suppose skel(μ) has more than four vertices. Then at least one of $\{a,b\}$, $\{b,c\}$, $\{c,d\}$, and $\{d,a\}$ is an inner edge. By Proposition 1, all of them are plane, and Proposition 2 therefore implies that skel(μ) has a separation pair. Hence, skel(μ) is not triconnected, a contradiction. As every vertex in a triconnected graph is incident to at least three edges, the skeleton is a K_4. □

Since a graph is planar if its triconnected components are planar, we obtain from Lemma 1:

Corollary 1 *Every o1p graph is planar.*

Let us take a closer look at *o1p* embeddings and the embedding of virtual edges. Here, we have to take into consideration that the expansion graph of every virtual edge e has at least one additional vertex besides the separation pair, so at least one segment of e must lie on the boundary of the outer face if the 2-clique-sums are to result in an *o1p* embedding. For an illustration consider the virtual edge $\{u,v\}$ in Fig. 2(b), whose expansion graph is a path of length three. The crossing edge $\{x,y\}$ partitions $\{u,v\}$ into two segments, hence, expg$(\{u,v\})$ must be embedded such that it replaces the edge segment of $\{u,v\}$ that lies in the outer face.

Proposition 3 *Every virtual edge must be embedded such that at least one segment is part of the boundary of the outer face.*

Combining this result with Lemma 1 and observing that every vertex must lie in the outer face (cf. Fig. 2(c)), we find:

Corollary 2 *The skeleton of every R-node must be embedded with exactly one pair of non-virtual, crossing edges and four plane edges.*

In contrast to *o1p* embeddings of graphs, a virtual edge may cross other incident edges. Consider again the graphs depicted in Fig. 2(b), e. g., and identify the vertices u and x in each graph such that the graphs contain a vertex "ux". Then, $\{ux,v\}$ and $\{ux,y\}$ are incident, crossing edges in the figure on top, but not after the virtual edge has been replaced by its expansion graph in the figure below. Assume for the time being that whenever an edge e' crosses a virtual edge $e=\{u,v\}$, then e' must cross all paths connecting u and v in expg(e).

For the following statement, note that the embedding of skeletons of S- and R-nodes, i. e., cycles and triconnected components, respectively, is not necessarily unique (S-node) or unique up to inversion (R-node) as in the planar case.

Lemma 2 *Let $e=\{u,v\}$ be a virtual edge in* skel(μ) *for a node μ in $\mathscr{T}$. If* refn(e) *is a P- or an R-node, then e must be embedded plane in* skel(μ). *If* refn(e) *is an S-node whose skeleton is the cycle $(u,c_1,c_2,\ldots,c_k,v,u)$, then e may cross at most one other edge, which must be virtual. In this case, e must be embedded such that the segment incident to u (v) lies in the outer face if the edge $\{u,c_1\}$ ($\{c_k,v\}$) is virtual.*

Proof First, consider the case where refn(e) is a P- or an R-node. Then, expg(e) is biconnected, so there are at least two vertex-disjoint paths from u to v in expg(e). Suppose e crosses another edge e' in skel(μ). If e' is non-virtual, then e' must cross both paths and therefore has at least two crossings. Otherwise, if e' is also virtual, there is at least one edge in $\mathrm{expg}(e')$ that crosses both paths or there is a vertex in $\mathrm{expg}(e')$ that is embedded such that it lies between these paths and therefore not in the outer face. This also applies if e crosses e' more than once. Consequently, e must always be embedded plane in $\mathrm{skel}(\mu)$.

Let now $\mathrm{refn}(e)$ be an S-node. As two S-nodes are never adjacent in $\mathscr{T}$, μ is either a P- or an R-node. By Corollary 2, however, e cannot be crossed as only plane edges may be virtual in the skeleton of an R-node. In the former case, the skeleton consists of vertices u and v and at least three parallel edges $\{u,v\}$, one of which is e. If e crosses two or more edges of these parallel edges, we again have the situation that a virtual edge crosses at least two vertex-disjoint paths, which cannot result in an *o1p* embedding of the graph, and likewise, if e crosses one edge more than once. Suppose e crosses only one edge e'. Let $\mathrm{skel}(\mathrm{refn}(e))$ be the cycle $(u,c_1,c_2,\ldots,c_k,v,u)$. As the vertices of $\mathrm{expg}(e)$ must lie in the outer face, this implies that when forming the 2-clique-sum of μ and $\mathrm{refn}(e)$, either $\{u,c_1\}$ or $\{c_k,v\}$ is crossed. Suppose $\{u,c_1\}$ is virtual. As $\mathrm{refn}(e)$ is an S-node, $\{u,c_1\}$ can only be refined by a P- or an R-node and therefore must be plane. Hence, the crossing must be at $\{c_k,v\}$. The same applies to $\{c_k,v\}$ with switched roles. Note that if e' is non-virtual, the *o1p* embedding of G has a pair of incident, crossing edges. □

Let us consider the situation where an edge e' crosses a virtual edge $e=\{u,v\}$ only partially, i. e., e' crosses only some, but not all paths connecting u and v in $\mathrm{expg}(e)$. Let p,q be two paths in $\mathrm{expg}(e)$ connecting u and v such that p is the first path crossed by e' and q is not crossed by e'. Note that p and q are not necessarily disjoint. Fig. 2(d) shows an example, where u might be either the leftmost vertex or the one to the right of it. If e' is virtual, it cannot be refined by a P- or R-node, by the same argument as in the first part of the proof of Lemma 2. Hence, e' is either refined by an S-node or non-virtual. In both cases, e' cannot cross p more than once without crossing a common section of p and q, so e' ends somewhere within $\mathrm{expg}(e)$ and is incident to either u or v. Let μ be the node of $\mathscr{T}$ whose skeleton contains both e and e'. Corollary 2 also applies for partial crossings, so μ cannot be an R-node. If μ is a P-node, u and v are the only vertices of $\mathrm{skel}(\mu)$ and $e'=\{u,v\}$. Furthermore, there must be a third (virtual or non-virtual) edge $\{u,v\}$ (drawn dotted in Fig. 2(d)). Together with this edge or, if it is virtual, its expension graph, no *o1p* embedding is possible. Consequently, μ must be an S-node and, as no two S-nodes are adjacent in $\mathscr{T}$, e' is non-virtual and $\mathrm{refn}(e)$ is either a P- or an R-node. This situation is indeed possible in an *o1p* embedding of a graph, as the example in Fig. 2(d) shows (without the dotted edge). Let $\nu=\mathrm{refn}(e)$. Then, the edge $\{u,v\}$ in $\mathrm{skel}(\nu)$ which is linked to e must be embedded with a crossing, because its expansion graph crosses other edges of $\mathrm{skel}(\nu)$. By Corollary 2, this implies that ν is a P-node and the crossing is represented adequately by a conventional, i. e., complete, crossing of the edge that is refined by μ in $\mathrm{skel}(\nu)$. In μ itself, we embed e without crossing. Consequently, it suffices for the remainder of this section to deal with "complete" crossings only.

As all edges in a P-node are parallel and every virtual edge may be crossed at most once, we obtain at most two pairs of crossing virtual edges, which then also form the boundary of the outer face. There may be a fifth, non-virtual edge that is completely inner. By summing up Corollary 2 and Lemma 2, and observing that an S-node can only be adjacent to P- and R-nodes in $\mathscr{T}$, we obtain:

Corollary 3 *Every virtual edge in the skeleton of an S-node must be embedded plane. The skeleton of every R-node contains at most four virtual edges, which must be embedded plane, and no vertex may be incident to more than two virtual edges. The skeleton of a P-node has at most four virtual edges.*

Algorithm 1 Recognition of *o1p*

```
procedure TestOuter1Planarity(G)
    if G is not planar then return ⊥                                        ▷ Corollary 1
    𝒯 ← SPQR-tree of G
    for all R- and P-nodes μ ∈ 𝒯 do
        if μ is an R-node then
            if skel(μ) ≠ K_4 or contains a vertex incident to > 2 virtual edges then
                return ⊥                                                    ▷ Lemma 1/ Corollary 3
            else
                for all neighbors ν of μ do
                    let e be the virtual edge in skel(μ) with refn(e) = ν and let e = {u,v}
                    if ν is an S- or an R-node then     ▷ if {u,v} is an edge of G, ν must be a P-node
                        insert a plane edge {u,v}                           ▷ Proposition 1
        else if μ is a P-node then           ▷ skeletons of P-nodes have exactly two vertices
            if skel(μ) contains > 4 virtual edges then return ⊥             ▷ Corollary 3
            else if μ has only virtual edges then insert a plane edge       ▷ Lemma 3
    compute the mapping 𝒞
    ℙ_F ← {fixable P-nodes}
    ℙ_N ← {P-nodes with crossings, but none fixable}
    while ℙ_F ∪ ℙ_N ≠ ∅ do
        while ℙ_F ≠ ∅ do
            remove next P-node π from ℙ_F with fixable S-nodes σ_1, σ_2
            z ← FixCrossingAtPNode(G, 𝒯, π, σ_1, σ_2)                      ▷ affected P-nodes
            if z = ⊥ then return ⊥
            for all π′ ∈ z do
                update 𝒞
                if π′ is fixable then move π′ from ℙ_N to ℙ_F
        if ℙ_N ≠ ∅ then                                                     ▷ Lemma 4
            choose any element π of ℙ_N with S-nodes σ_1, σ_2 conformant to 𝒞
            z ← FixCrossingAtPNode(G, 𝒯, π, σ_1, σ_2)
            for all π′ ∈ z do
                update 𝒞
                if π′ is fixable then move π′ from ℙ_N to ℙ_F
    for all S-/P-/R-nodes μ ∈ 𝒯 do fix the embedding
    return 2-clique-sum of the skeleton embeddings
```

With these findings, we are ready for the *o1p* recognition algorithm (Algorithm 1). By Corollary 1, *o1p* graphs are planar. The algorithm uses this as a prerequisite and computes the SPQR-tree of the input graph. Both subroutines take $\mathcal{O}(n)$ time [29] and the number of nodes in $\mathcal{T}$ for a planar graph is always in $\mathcal{O}(n)$. During the following steps, we augment G by adding edges, which are plane in all *o1p* embeddings of G. The conditions for R-nodes can be checked in time $\mathcal{O}(1)$ per R-node. Additionally, if an R-node is adjacent to another R-node or an S-node, then one of the edges of K_4 is missing. As an example, see the R-nodes ρ_1 and ρ_2 in Fig. 3(b). By Proposition 1, however, the edge may be inserted and is always plane. Observe that this introduces a new P-node π_5 in Fig. 3(c). As an R-node may have at most four neighbors and as the SPQR-tree can be updated in $\mathcal{O}(1)$ time, this modification takes constant time, too.

The following lemma allows us to insert a non-virtual edge in every P-node if there is none. In Fig. 3(b), this would apply, e. g., to π_1.

Lemma 3 *Let u, v be the vertices in the skeleton of a P-node without non-virtual edges. Then the insertion of the edge $\{u,v\}$ does not violate outer 1-planarity and $\{u,v\}$ is plane for every o1p embedding of G.*

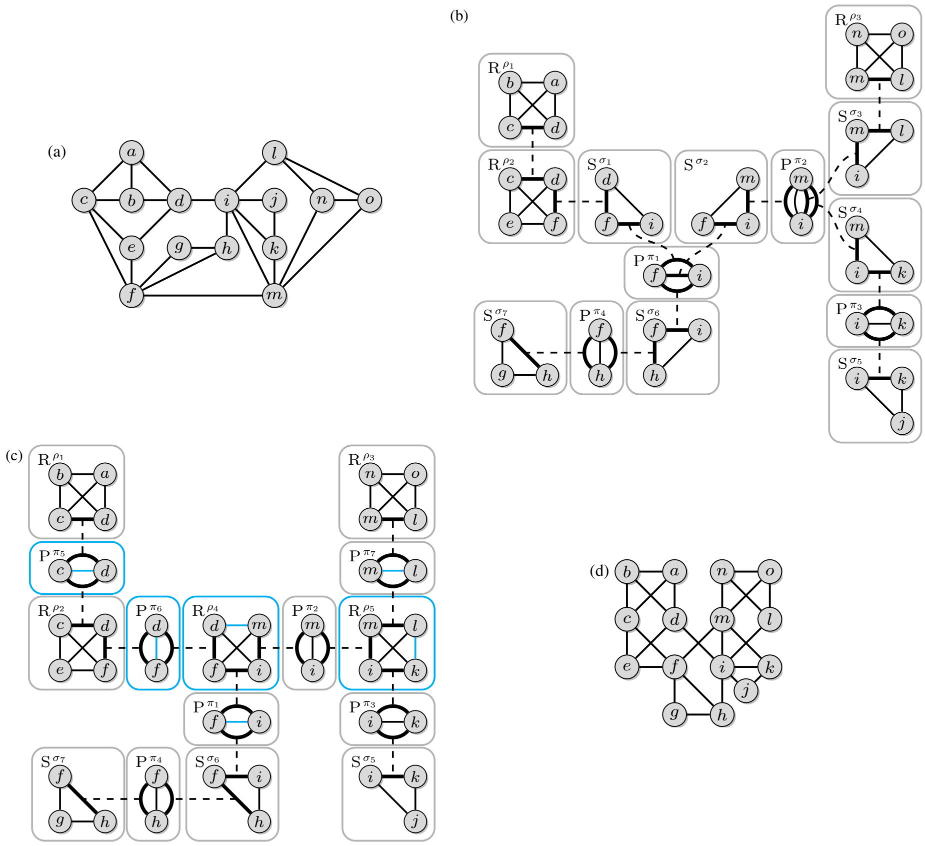


Fig. 3: Input graph (a), its SPQR-tree (b), the SPQR-tree after the algorithm (c) (new edges and nodes colored), and the found *o1p* embedding (d).

Proof Let π be a P-node whose skeleton has vertices u, v that are connected by virtual edges only. According to the definition of SPQR-trees, every skeleton of a P-node has at least three edges. Hence, π is adjacent to at least three other nodes. Subsequently, at least two virtual edges must be refined by S-nodes and are embedded with a crossing. This results in a crossing of two non-virtual edges in G that are, by Lemma 2, incident to u and v, respectively. By Proposition 1, the edge $\{u, v\}$ can always be inserted and is plane. $\square$

Again, Algorithm 1 can check these two conditions and augment the graph for a P-node in time $\mathcal{O}(1)$, which results in a running time of $\mathcal{O}(n)$ for lines 4–15.

Consider a P-node π with vertices u, v. If $\text{skel}(\pi)$ has at most two virtual edges, they can be embedded without crossing and such that both lie completely in the outer face. Other embeddings may be possible, but they result in unnecessary crossings. Suppose $\text{skel}(\pi)$ has at least three virtual edges. In consequence of Proposition 3, two of them must cross each other. In Fig. 3(b), this holds for π_1 and π_2. We say that a P-node π *claims* a non-virtual edge e, and express this by defining the mapping $\mathscr{C}(e) = \pi$, if e is crossed in every embedding of $\text{skel}(\pi)$ that conforms with Lemma 2. Observe that $\mathscr{C}$ is uniquely defined, since G is *o1p* and thus, no edge may be crossed more than once. We say that an embedding of the skeleton of a P-node is *admissible* if the embedding of every edge conforms with Lemma 2 and does not

Algorithm 2 Fix the embedding of a P-node with two crossing S-nodes

1: **function** FIXCROSSINGATPNODE(G, $\mathcal{T}$, P-Node π, S-Node σ_1, S-Node σ_2)
2: let u, v be the separation pair of π
3: let $(u, c_1, \ldots, c_k, v, u)$ be the cycle in $\mathrm{skel}(\sigma_1)$
4: let $(u, d_1, \ldots, d_l, v, u)$ be the cycle in $\mathrm{skel}(\sigma_2)$
5: **if** edge $\{c_k, v\}$ is virtual **or** edge $\{u, d_1\}$ is virtual **then**
6: **if** edge $\{u, c_1\}$ is virtual **or** edge $\{d_l, v\}$ is virtual **then return** $\perp$ ▷ Lemma 2
7: **else** swap the roles of σ_1, σ_2
8: $\mathbb{P}_d \leftarrow \emptyset$ ▷ possibly affected P-nodes
9: **if** $k > 1$ **then**
10: insert edge $\{u, c_k\}$ in G, update $\mathcal{T}$
11: **if** $\{c_{k-1}, c_k\}$ is virtual **then** add its associated P-node to $\mathbb{P}_d$
12: **else if** $\{u, c_k\}$ is virtual **then** add its associated P-node to $\mathbb{P}_d$
13: **if** $l > 1$ **then**
14: insert edge $\{v, d_1\}$ in G, update $\mathcal{T}$
15: **if** $\{d_1, d_2\}$ is virtual **then** add its associated P-node to $\mathbb{P}_d$
16: **else if** $\{v, d_1\}$ is virtual **then** add its associated P-node to $\mathbb{P}_d$
17: insert edge $\{c_k, d_1\}$, update $\mathcal{T}$
18: **if** π has two (other) virtual edges **then** add π to $\mathbb{P}_d$
19: **return** $\mathbb{P}_d$

imply the crossing of non-virtual edges claimed by other P-nodes. In Fig. 3(b), e. g., $\mathrm{skel}(\pi_1)$ has two admissible embeddings and both imply crossing the edge $\{f, m\}$, either by $\{d, i\}$ or by $\{h, i\}$. Hence, π_1 claims $\{f, m\}$. Computing $\mathscr{C}$ involves checking the embeddings of the skeletons of all P-nodes. As every P-node has at most four virtual edges, there are at most $\binom{4}{2} \cdot 2 = 12$ embeddings. Hence, the total time needed for this step is in $\mathcal{O}(n)$.

If every admissible embedding of $\mathrm{skel}(\pi)$ yields the same set of edges that are crossed, then π is called *fixable*. Let e, e' be two virtual edges that are embedded crossing each other. Observe that in this case, two S-nodes, namely $\mathrm{refn}(e)$ and $\mathrm{refn}(e')$, are "crossing". By Proposition 1, the crossing can be augmented to a K_4. The insertion of these additional edges transforms each crossing S-node into an R-node that represents K_4. In Fig. 3(b), this occurs at π_1, σ_1, and σ_2. If the skeleton of an S-node previously had exactly three vertices, it is now completely contained in a K_4. Otherwise, the number of its vertices is reduced by exactly one, i. e., the vertex u or v, respectively. Note that completing a K_4 may affect the number of admissible embeddings, and, hence the fixability of other P-nodes if there was an admissible embedding of their skeletons that implied crossing one of e or e'. Algorithm 2 checks whether or not the virtual edges may cross each other (lines 5–7) and fixes the embedding of π (line 17). In order to ensure a linear running time of Algorithm 1, Algorithm 2 returns the set of affected P-nodes, i. e., the set of P-nodes whose number of admissible embeddings may have been reduced and, hence, which must be reconsidered in Algorithm 1.

The next lemma enables us to proceed, even if there is no fixable P-node.

Lemma 4 *Let π be a non-fixable P-node. If $\mathcal{T}$ has no fixable P-nodes, then every admissible embedding of* $\mathrm{skel}(\pi)$ *maintains at least one admissible embedding for every other P-node.*

Proof Consider the fixing procedure of an embedding for a P-node π and S-nodes σ' and σ''. Let e' and e'' be the non-virtual edges that are crossed thereby. This affects the number of admissible embeddings for the skeletons of at most two other P-nodes π' and π'' that are adjacent to σ' and σ'', respectively. Observe that $\pi' \neq \pi''$, as $\mathscr{T}$ is a tree, and that every non-virtual edge is represented in the skeleton of exactly one node of $\mathscr{T}$.

Consider π'. W. l. o. g., let e' be the non-virtual edge in skel(σ') that is crossed after the fixing. Then the number of admissible embeddings of skel(π') is reduced by exactly those that implied crossing e', too. However, π' did not claim e', so there is at least one other admissible embedding of skel(π'). The same argument holds for π'' and e''. □

Hence, by applying Lemma 4, we can step by step fix all embeddings of the skeletons of P-nodes with at least three virtual edges. Thereafter, every P-node has exactly two virtual edges and one non-virtual (cf. Fig. 3(c)). In Algorithm 1, this corresponds to lines 19–32. FIXCROSSINGATPNODE takes $\mathscr{O}(1)$ time per call and there are embeddings of at most $\mathscr{O}(n)$ P-nodes to fix. Hence, the time for this part is $\mathscr{O}(n)$. The algorithm concludes by selecting an admissible embedding for all P- and R-nodes. All remaining S-nodes are embedded as plane cycles. An embedding of G can be obtained via the 2-clique-sums of all skeleton embeddings (cf. Fig. 3(d)). Consequently, Algorithm 1 has a running time of $\mathscr{O}(n)$.

It remains to show that all conditions presented so far are also sufficient for a graph to be *o1p*. Every skeleton is, taken by itself, embedded *o1p*. Let $\{u,v\}$ be a virtual edge that links two nodes μ and ν and consider their 2-clique-sum skel$(\mu) \oplus$ skel(ν). After the augmentation of Algorithm 1, every virtual edge is embedded such that it lies completely on the boundary of the outer face. In the embeddings of μ, let o_μ and i_μ be the faces on either side of $\{u,v\}$ such that o_μ is the outer face, and define o_ν and i_ν analogously in the embedding of ν. As every skeleton is biconnected, $o_\mu \neq i_\mu$ and $o_\nu \neq i_\nu$. For the *o1p* embedding of skel$(\mu) \oplus$ skel(ν), combine both skeleton embeddings such that o_μ and o_ν are joined as well as i_μ and i_ν. This may require to invert one of both skeleton embeddings, i. e., for every face, the cyclic sequence of edges that forms its boundary is reversed. In the resulting embedding, every vertex then still lies in the outer face and every edge is crossed at most once. The outer 1-planarity of the whole embedding follows by induction. We can summarize:

Lemma 5 *A graph G is o1p if and only if it is a subgraph of a graph H with SPQR-tree $\mathscr{T}$ such that R-nodes and S-nodes are adjacent to P-nodes only, every skeleton of an R-node is a K_4, and every skeleton of a P-node has exactly one non-virtual and two virtual edges.*

This concludes the proof of Theorem 1. Additionally, if a graph is *o1p*, Algorithm 1 provides an *o1p* embedding. With little extra effort, we can augment G to maximality. Consider the supergraph H constructed from G by Algorithm 1 and its SPQR-tree. It may have S-nodes with four or more vertices. As all remaining S-nodes are embedded plane, we can insert a plane edge between two non-adjacent vertices, which splits the S-node into two smaller S-nodes with an intermediate P-node. This procedure can be repeated until all S-nodes are triangles. Next, consider a P-node that is adjacent to exactly two S-nodes, e. g., π_4 in Fig. 3(c). Then we can insert a crossing edge ($\{g,i\}$ in the example) that augments the subgraph to K_4. As a result, the nodes π_4, σ_6, and σ_7 are replaced by a new R-node. Observe that this is the only part where the augmentation introduces a new crossing. We denote this supergraph of H by H^+. Its SPQR-tree consists of R-nodes, each of which corresponds to K_4 and of S-nodes each of which corresponds to a triangle. R- and S-nodes are only connected via P-nodes, which in turn have exactly two virtual edges and one non-virtual edge. Consider an embedding of H^+. It has a tree-like structure that consists of K_4s and triangles (K_3s) that share an edge if and only if their corresponding R- and S-nodes are connected via a P-node. As no P-node is adjacent to two S-nodes, triangles can only share an edge with K_4s. Suppose H^+ was not maximal. If we were able to insert an inner, plane edge, this would

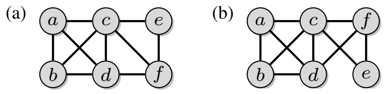


Fig. 4: (a) A plane-maximal ø1p embedding. (b) After flipping the vertices e and f the plane edge (d,e) can be added.

correspond to inserting a P-node into the SPQR-tree of H^+. However, no two P-nodes can be adjacent. Inserting an inner, crossed edge is equal to augmenting two triangles to a K_4, which is impossible, too, as no P-node is adjacent to two S-nodes. Finally, consider adding an edge to the outer face. As every crossing has been augmented to a K_4, the boundary of the outer face consists of a plane Hamiltonian cycle. Hence, every additional edge would separate at least one vertex from the outer face. Consequently, we can easily extend Algorithm 1 such that it maximizes the input graph. Additionally, we obtain another characterization:

Lemma 6 *A graph G is maximal o1p if and only if the conditions for H in Lemma 5 hold and no P-node in its SPQR-tree is adjacent to more than one S-node and the skeleton of every S-node is a cycle of length three.*

The argument above also implies that every embedded maximal *o1p* graph is maximal for all *o1p* embeddings. By contrast, this does neither hold for embedded maximal 1-planar graphs [12], nor for embedded plane-maximal *o1p* graphs as shown in Fig. 4.

Corollary 4 *A graph G is maximal o1p if it has a maximal o1p embedding.*

Due to Lemma 6, the embedding of a maximal *o1p* graph is fixed if and only if the embedding of the skeleton of every R-node is fixed. This, in turn, is the case if and only if it contains at least two incident virtual edges.

Corollary 5 *The embedding of a maximal o1p graph is unique up to inversion if and only if the skeleton of every R-node of its SPQR-tree contains a vertex that is incident to exactly two virtual edges.*

A plane-maximal *o1p* graph is obtained if the step that augments a P-node with two adjacent triangle S-nodes to a K_4 is omitted. In the same way we can adjust Algorithm 1 to test (plane) *o1p* maximality.

Corollary 6 *There is a linear-time algorithm to test whether a graph is maximal (plane-maximal) o1p and to augment an o1p graph to a maximal (plane-maximal) o1p graph.*

3.2 Minors of non-*o1p* Graphs

From the recognition algorithm, we can immediately derive minors of non-*o1p* graphs: If the algorithm returns $\perp$, the graph at hand contains at least one of the *o1p* minors M as depicted in Fig. 5.

However, the *o1p* minors cannot be used for a characterization of *o1p*, since *o1p* ist not closed under edge contraction and subdivision, cf. Sect. 4.2. For instance, Fig. 6 shows an *o1p* graph where contracting the dashed edge yields $K_{2,5}$. Hence, the converse of Theorem 2 does not hold true.

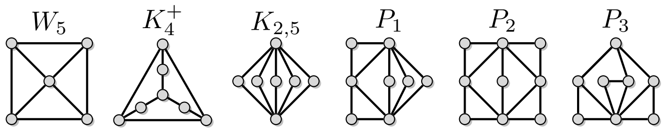


Fig. 5: The set M of minors of non-*o1p* graphs

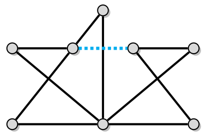


Fig. 6: An *o1p* graph that contains $K_{2,5}$ as a minor.

Theorem 2 *If a graph is not o1p, it contains at least one graph in M as a minor. Further, M is minimal and every graph in M is not o1p while removing or contracting an edge makes it o1p.*

Proof Observe that removing or contracting an edge of a graph in M yields an *o1p* graph and none of them is the minor of another. We show that every member of M is not *o1p* and that every non-*o1p* graph contains a graph in M as a minor.

As a general observation, note that the skeleton of a node in an SPQR-tree $\mathscr{T}$ of a graph G is by itself a minor of G. Let $\{u,v\}$ be a virtual edge of a node of $\mathscr{T}$. Recall that we omit Q-nodes and represent edges directly in the skeletons, i. e., an edge is either virtual and refined by an S-, P-, or R-node or it is non-virtual. The expansion graph of $\{u,v\}$ contains at least one vertex w distinct from u and v, and a minor of the expansion graph is the path u,w,v of length two. In our proof, we use this observation to replace any virtual edge by a path of length two to obtain our minors of M. In a nutshell, this path captures the principle structure behind the virtual edge $\{u,v\}$ which makes it impossible to obtain an *o1p* embedding.

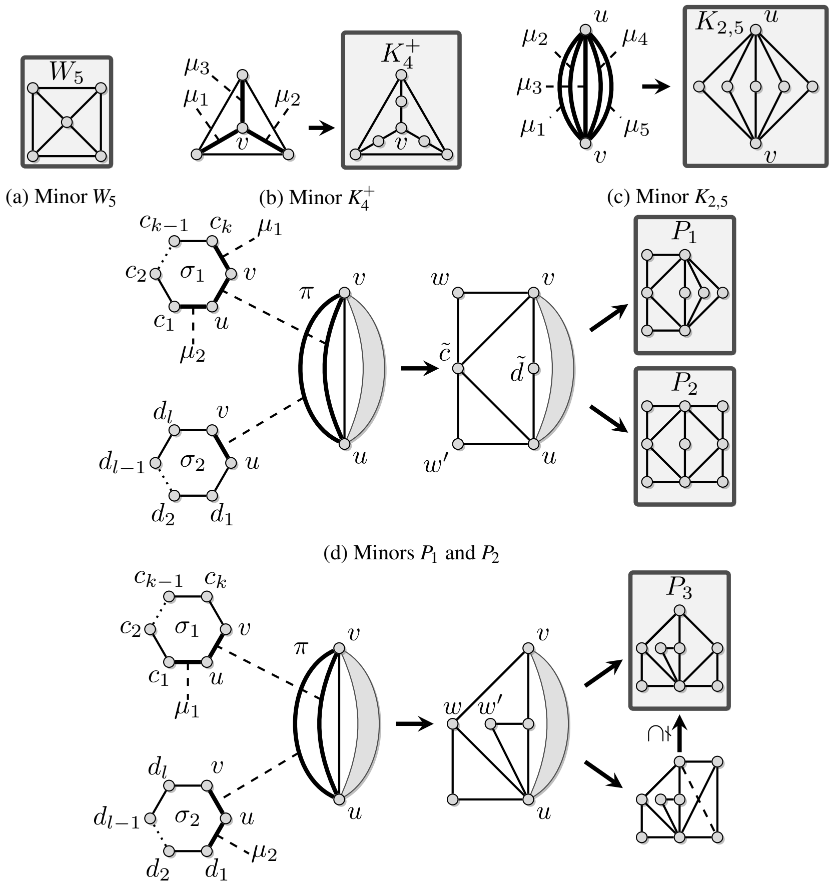
The proof is completed by a case differentiation on all lines of Algorithms 1 and 2, where $\perp$ is returned.

Line 2 in Algorithm 1 (W_5) The algorithm returns $\perp$ if G is not planar. In this case, G contains K_5 or $K_{3,3}$ as a minor. W_5 is a subgraph of K_5 and thus also a minor. Further, as every vertex of $K_{3,3}$ has degree three, contracting one of its edges yields a graph with five vertices, where one vertex has degree four and all others have degree three. Hence, we have obtained W_5, which is not *o1p* by Lemma 1.

Line 6 in Algorithm 1 (W_5 and K_4^+) We have an R-node ρ whose skeleton $H = \text{skel}(\rho)$ is triconnected and planar. $\perp$ is returned if H contains more than four vertices (**Case 1**) or at least one vertex which is incident to three or more virtual edges (**Case 2**).

Case 1: First, suppose H has exactly five vertices. We show that H must contain W_5 as a subgraph. As H is triconnected, every vertex in H is incident to at least three edges, i. e., H has at least eight edges. Therefore, H consists of one vertex x of degree at least four and a set Y of four vertices of degree at least three. As there are only five vertices in total, x is adjacent to all four vertices in Y and each vertex in Y is adjacent to at least two other, distinct vertices in Y. Hence, H contains W_5 as a subgraph.

In a variation of Tutte's Wheel Theorem, Thomassen [41] showed that every triconnected graph on at least five vertices contains an edge such that contracting this edge and



(a) Minor W_5 (b) Minor K_4^+ (c) Minor $K_{2,5}$

(d) Minors P_1 and P_2

(e) Minors P_3

Fig. 7: Different cases in the proof of Theorem 2

replacing multi-edges by single edges yields a triconnected graph. In consequence, every triconnected graph on at least six vertices can be contracted to a triconnected graph on five vertices, which in turn contains W_5, i. e., W_5 is a minor of all these graphs.

Case 2: In this case, H contains a vertex v which is incident to three or more virtual edges (see Fig. 7(b)). We assume that H contains at most four vertices, otherwise **Case 1** applies. As H is triconnected, H equals K_4. In consequence, v is incident to exactly three virtual edges. Replacing these virtual edges by paths of length two and all other virtual edges by simple edges, we obtain K_4^+ as a minor as depicted in Fig. 7(b). K_4^+ is not *o1p* since its SPQR-tree contains an R-node with a vertex incident to three virtual edges, which violates *o1p* by Corollary 3.

Line 14 in Algorithm 1 ($K_{2,5}$) In this case, we have a P-node with at least five virtual edges (see Fig. 7(c)). By replacing five virtual edges by a path of length two and removing all other edges, we obtain $K_{2,5}$ as a minor. Again, $K_{2,5}$ cannot be *o1p* since it contains the P-node with five virtual edges from which it is derived (cf. Corollary 3).

Line 6 in Algorithm 2 (P_1, P_2, and P_3) In this case, we have a P-node π with vertices u and v to which two virtual edges e_1, e_2 are incident. The edges e_1 and e_2 are refined by two S-nodes σ_1 and σ_2, respectively. The skeleton skel(σ_1) consists of the cycle $(u, c_1, \ldots, c_k, v, u)$ and the skeleton skel(σ_2) of the cycle $(u, d_1, \ldots, d_l, v, u)$. $\perp$ is returned if $\{c_k, v\}$ or $\{u, d_1\}$ is virtual and if additionally $\{u, c_1\}$ or $\{d_l, v\}$ is virtual, which results in four cases.

In the first case, $\{c_k, v\}$ and $\{u, c_1\}$ are virtual. This situation is depicted on the left side of Fig. 7(d). Note that the case, where $\{u, d_1\}$ and $\{d_l, v\}$ are virtual, is symmetric and, hence, the following observations hold equally. In the skeleton of P-node π, there are two virtual edges to the left separated by the non-virtual edge $\{u, v\}$ from the right part which is sketched by the shaded region. The virtual edges $\{c_k, v\}$ and $\{u, c_1\}$ are refined by the nodes μ_1 and μ_2, respectively. Since σ_1 is an S-node, each of μ_1 and μ_2 is either a P- or an R-node. Hence, a minor of expg$(\{c_k, v\})$ is the triangle consisting of c_k, v, and w and, likewise, a minor of expg$(\{u, c_1\})$ is the triangle u, c_1, and w'. Further, the edges of path $c_1, c_2, \ldots, c_{k-1}, c_k$ can be contracted until c_1 and c_k are identified and replaced by vertex $\tilde{c}$. The virtual edge e_2 is replaced by a path of length two. Altogether, we obtain the graph as shown in the middle of Fig. 7(d). By Lemma 2, e_1 must be embedded such that the segment incident to v as well as the segment incident to v must lie in the outer face, which implies that e_1 must be embedded plane in skel(π). However, e_2 may be crossed. Next, we investigate the possibilities for the right part of π.

In order to force the situation on the left hand side, there are several possibilities for the shaded region: As e_1 may not be crossed, two additional virtual edges e_3, e_4 that correspond to an S-node each would suffice: By Proposition 3, every virtual edge must lie with at least one segment in the outer face and by Lemma 2, every virtual edge may be crossed at most once. As e_1 is plane and must lie in the outer face, not all three virtual edges e_2, e_3, e_4 can have a segment in the outer face, too. By replacing both e_3 and e_4 by paths of length two, we thus obtain P_1. P_1 is not *o1p* since its SPQR-tree violates Lemma 2 and Proposition 3 as just described. Otherwise, the shaded region contains only one virtual edge e_3 that must be embedded plane. Then again by Lemma 2, e_3 is either refined by an S-node that has the same structure as S-node σ_1 or it is refined by an R-node. In the latter case, we again obtain P_1 as a minor. In the former case, skel(π) contains two virtual edges e_1 and e_3 which must both be embedded plane (cf. Lemma 2) and in the outer face (cf. Proposition 3), so e_2 cannot be embedded such that at least one segment lies in the outer face, as Proposition 3 also requires. This yields P_2, which is non-*o1p* by this reasoning. Note that if the region contains more than two virtual edges, then $K_{2,5}$ is a minor as discussed previously.

In the second case, $\{u, d_1\}$ and $\{u, c_1\}$ are virtual (see Fig. 7(e)). Again, the reasoning is the same for the symmetric case where $\{d_l, v\}$ and $\{c_k, v\}$ are virtual. Let e_1, e_2 be the two virtual edges of π on the left side which are refined by the S-nodes σ_1 and σ_2, respectively. The right hand side is again sketched as a shaded region. As before, we can replace the virtual edges $\{u, d_1\}$ and $\{u, c_1\}$ by the triangles u, w, d_1 and u, w', c_1, respectively. Further, we contract the edges of remaining parts in the skeleton of the S-nodes until a single edge is left. The resulting graph is shown in the middle of Fig. 7(e). In this case, the skeleton of P-node π has no admissible embedding already for a single virtual edge on the right hand side: Both e_1 and e_2 require that their segment incident to u lies in the outer face. If there is a virtual edge e_3 on the right hand side with the same requirement, then skel(π) has no admissible embedding as no pair of these three virtual edges may cross each other (cf. Lemma 2). Note that if there are up to two virtual edges on the right hand side that allow a crossing such that their segment incident to u lies not in the outer face, skel(π) has an admissible embedding. Edge e_3 is either refined by an S- or by an R-node. In case of an S-node, we obtain the same situation as with σ_1 and σ_2 and this yields P_3. If e is refined by an R-node, then a minor of $\text{expg}(e)$ is K_4 with one edge removed (see bottom right of Fig. 7(e)). However, by removing the dashed edge, we obtain P_3. Hence, for both cases, the minor is P_3 and P_3 is non-*o1p* by this reasoning. □

4 Properties of Outer 1-Planar Graphs

We now establish several structural properties of *o1p* graphs, which show that they are closer to outerplanar than to planar graphs.

4.1 Density

(Maximal) biconnected outerplanar graphs are characterized as planar graphs whose weak duals are (binary) trees. In weak duals, the outer face is ignored. We extend the notion of the dual tree to maximal *o1p* embeddings as follows.

Let G be a graph with a fixed maximal *o1p* embedding. The vertices incident to every pair of crossing edges induce a K_4 in G by Proposition 1. Let $\bar{G}$ be obtained from G by removing each such pair of crossing edges. Then $\bar{G}$ is outerplanar and the inner faces are triangles or quadrangles. The weak dual $\bar{G}^*$ is a tree and we distinguish two types of nodes. $\triangle$*-nodes* correspond to triangles and $\boxtimes$*-nodes* correspond to quadrangles, which arose from erasing the pair of crossing edges from the K_4s. Accordingly, these vertices have degree at most 3 and 4, respectively. Two nodes are adjacent in $\bar{G}^*$ if and only if the corresponding faces in $\bar{G}$ (or, equivalently, cliques in G) have an edge in common. We refer to $\bar{G}^*$ as the *dual tree of* G in order to emphasize the structural similarity between outerplanar and *o1p* graphs.

The dual tree can be obtained directly from the SPQR-tree $\mathscr{T}$ of G: By Lemma 6, the skeleton of every R-node of $\mathscr{T}$ is a K_4 and the skeleton of every S-node is a triangle, which correspond one-to-one to $\boxtimes$-nodes and $\triangle$-nodes, respectively. Furthermore, every P-node has exactly two virtual edges. By "skipping" the P-nodes and connecting the $\boxtimes$- and $\triangle$-nodes directly, we obtain $\bar{G}^*$. From Lemma 6 we conclude:

Corollary 7 *The weak dual of a maximal o1p graph consists of $\triangle$- and $\boxtimes$-nodes such that two $\triangle$-nodes are not adjacent.*

The dual tree has many nice properties. First, it allows us to analyze the density of maximal *o1p* graphs and establish tight upper and lower bounds. More specifically, we can express the number of vertices and edges of G by seeing G as a 2-clique-sum of its K_3s and K_4s.

Lemma 7 *The number of vertices of a maximal o1p graph G is $n = N_3 + 2N_4 + 2$ and the number of edges is $2n + N_4 - 3$ where N_3 and N_4 are the numbers of $\triangle$- and $\boxtimes$-nodes in $\bar{G}^*$.*

Proof Let $N = N_3 + N_4$. When summing up the number of vertices and edges in each clique one has to subtract two vertices and one edge counted twice as they are shared by the cliques of adjacent nodes. This happens $N-1$ times as the tree $\bar{G}^*$ has $N-1$ edges. Hence, the number of vertices and edges of G is $3N_3 + 4N_4 - 2(N-1)$ and $3N_3 + 6N_4 - (N-1)$, respectively. □

From the dual tree and the bounds established in Lemma 7 we can easily derive upper and lower bounds on the density of *o1p* graphs and construct graphs which show that the bounds are tight. The upper bound was also proved by Didimo [20] using a different approach.

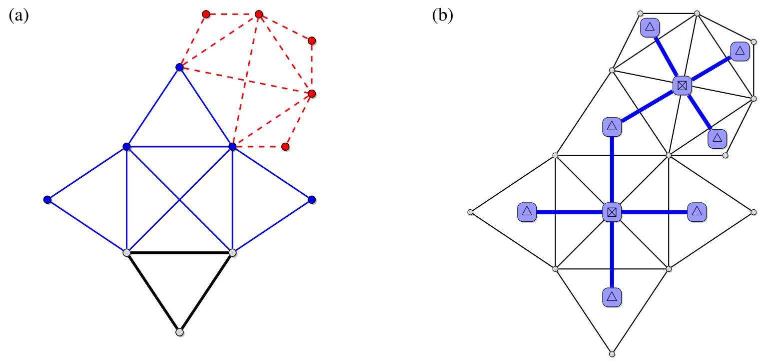


Fig. 8: (a) Example of a sparsest maximal $o1p$ graph as constructed in Theorem 4. The construction begins with the thick, black K_3. Construction step (ii) is applied twice, resulting in the addition of the solid blue and then of the red dashed 5 vertices and 11 edges. (b) The same graph along with its dual tree.

Theorem 3 *An $o1p$ graph with n vertices has at most $\frac{5}{2}n-4$ edges, and for every even $n \geq 2$ there are $o1p$ graphs with $\frac{5}{2}n-4$ edges.*

Proof The densest maximal $o1p$ graph G with at least four vertices has a dual tree solely consisting of ⊠-nodes, which by Lemma 7 results in graphs with $\frac{5}{2}n-4$ edges. And for every even $n \geq 2$ such graphs can be constructed, e. g., as a chain of K_4s, see also Fig. 1(a).

Theorem 4 *Every maximal $o1p$ graph with n vertices has at least $\frac{11}{5}n - \frac{18}{5}$ edges, and for every $k \geq 0$ and $n = 5k+3$ this bound is tight.*

Proof A maximal $o1p$ graph G is sparsest if the share of ⊠-nodes is minimal. However, two △-nodes cannot be adjacent in $\bar{G}^*$ by Corollary 7. If two ⊠-nodes are adjacent, one can obtain a sparser graph which corresponds to inserting a △-node in between. Similarly, if a ⊠-node is adjacent to less than four neighbors, G can be sparsified by attaching another △-node. Thus, in the dual tree of the sparsest maximal $o1p$ graphs, every ⊠-node is adjacent to exactly four △-nodes and all leaves are △-nodes. These graphs can be constructed by (i) starting with a △-node and repeatedly (ii) selecting any △-node with less than three neighbors and attaching a ⊠-node adjacent to another three △-nodes. Figure 8(a) shows a maximal $o1p$ graph where construction step (ii) has been applied twice. When construction step (ii) is applied k times, the resulting graph G consists of $5k+3$ vertices and $11k+3$ edges. □

These results contribute to investigations on the density of maximal graphs, including planar graphs with exactly $3n-6$ edges, outerplanar graphs with $2n-3$ edges, and 1-planar graphs with a range from $\frac{21}{10}n - \mathcal{O}(1)$ to $4n-8$ and known sparse maximal 1-planar graphs with $\frac{45}{17}n + \mathcal{O}(1)$ edges [12].

Corollary 8 *If G is a maximal $o1p$ graph of size n, then G has m edges with $\frac{11}{5}n - \frac{18}{5} \leq m \leq \frac{5}{2}n-4$ and these upper and lower bounds are tight.*

4.2 Edge Contraction

Recall that a graph H is a minor of a graph G if H is obtained from G by vertex and edge deletions and edge contractions. Conversely, a subdivision splits an edge into two by placing

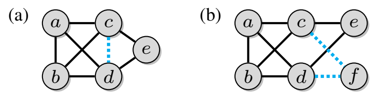


Fig. 9: (a) $\{c,d\}$ is inner in every *o1p* embedding. (b) *o1p* embedding with subdivision of $\{c,d\}$ by f.

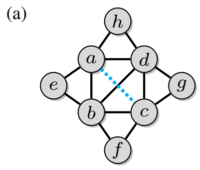


Fig. 10: (a) $\{a,c\}$ is crossed in every *o1p* embedding. (b) *o1p* embedding after contraction of $\{a,c\}$.

a vertex on it. It is well-known that the planar and the outerplanar graphs are closed under taking minors, which means applying the respective operations. Also the subdivision of a (non-) planar graph is (non-) planar, whereas subdivisions may destroy outerplanarity.

1-planar graphs are not closed under taking minors, since every graph is a subdivision of a 1-planar graph. However, the subdivision of a 1-planar graph is 1-planar. Both edge contractions and subdivisions may destroy outer 1-planarity.

Theorem 5 *Let G be an embedded o1p graph. If H is obtained from a subgraph of G by contractions of plane edges, then H is o1p. Also, H is o1p if it is obtained by subdivisions of outer edges.*

Proof The contraction of a plane edge $\{u,v\}$ of an embedded *o1p* (1-planar) graph does not induce more crossings per edge. This can be obtained immediately from the planarization with special vertices for the crossing points. However, the edge contraction may introduce crossing among incident edges, which can be untangled. Clearly, all vertices remain in the outer face, as they do by subdividing an outer edge. Obviously, taking subgraphs preserves *o1p*. □

Sometimes, even crossed edges can be contracted and inner edges can be subdivided while preserving *o1p*, as illustrated in Figs. 9 and 10. However, contracting the dashed edge of the graph in Fig. 6 yields $K_{2,5}$ as a minor and destroys *o1p*. What is legal? This can be decided using our linear-time recognition algorithm for the resulting graph.

Corollary 9 *There is a linear time algorithm to test whether or not an edge of an o1p graph can be contracted (subdivided) such that the resulting graph is o1p.*

4.3 Parameters

Next, we consider some classical graph parameters which increase by one from outerplanar to *o1p* graphs. In order to obtain upper bounds on these parameters, we consider maximal *o1p* graphs where it suffices to assume biconnectivity. For each biconnected graph the treewidth is two if and only if the SPQR tree has only S- and P-nodes, and it is three if it contains an R-node by Lemma 1. Thus, we can directly compute the treewidth of *o1p* graphs in linear time, which can also be obtained from Bodlaender's theorem [11]. Since maximal *o1p* graphs are chordal, the treewidth equals the clique size minus one [13].

Corollary 10 *Every o1p graph has treewidth at most three and this bound is tight. The treewidth of an o1p graph and the tree-decomposition can be computed in linear time.*

A *stack (queue) layout* of a graph consists of a total order of the vertices and a partition of the edges into stacks (queues), such that no two edges in the same stack (queue) are intersecting (nested) [8, 21]. Bernhart and Kainen [8] have characterized the outerplanar graphs as the 1-stack graphs, and the 2-stack graphs as the subgraphs of planar graphs with a Hamiltonian cycle.

Theorem 6 *Every o1p graph has a stack number of at most two, and this bound is tight. The stack number of an o1p graph can be computed in linear time.*

Proof First, we show that two stacks suffice. Augment the graph to be maximal *o1p* and choose an *o1p* embedding. Fix the order of the vertices for the 2-stack-layout by starting with an arbitrary vertex and then following the order of the vertices in the unbounded face. Color the edges red and blue such that edges of the same color do not cross. Then the red and the blue graphs for their own are embedded outerplanar, thus each has a 1-stack-layout which adheres to the same prescribed order.

For the tightness of the bound recall that K_4 is not outerplanar, thus actually requires two stacks. For the computation consider each biconnected component separately and take the maximum. The stack number is one if the graph is outerplanar. □

Let us turn to the queue number. Every outerplanar graph has queue number two [30], and every planar graph has a polylogarithmic queue number [18]. The fact that *o1p* graphs have bounded queue number follows from a result of Dujmović et al. [21] on graphs with bounded treewidth. In fact, the queue number increases to three for *o1p* graphs, and there are *o1p* graphs with queue number two, which are not outerplanar, such as K_4.

Theorem 7 *Every o1p graph has a queue number of at most three, and this bound is tight.*

Proof First, we show that three queues suffice. Consider an embedded maximal *o1p* graph G. Run a breadth-first search (BFS), which starts from an arbitrary vertex r and visits the vertices in accordance with the embedding. BFS assigns each vertex v a BFS number $\mathrm{bfs}(v)$, which determines the order of the vertices in the 3-queue-layout.

The vertices are partitioned into levels according to their distance from r. Then we distinguish two types of edges. *Inter-level* edges span adjacent levels while *intra-level* edges connect vertices of the same level. Color the inter-level edges blue and red such that crossing edges receive different colors. The situation is exemplified in Fig. 11. The blue graph on its own is embedded proper-leveled planar. That means, it can be processed in a single queue [30] and the same holds for the red graph. It remains to show that the intra-level edges can be processed in a third queue. Consider two adjacent vertices u and v on the same level. Assume there is another vertex w on this level such that $\mathrm{bfs}(u) < \mathrm{bfs}(w) < \mathrm{bfs}(v)$. There must be an inter-level edge e connecting w from some vertex on the previous level. Since the BFS adhered to the *o1p* embedding, $\{u, v\}$ and e cross. Hence, there can be at most one such w and $|\mathrm{bfs}(v) - \mathrm{bfs}(u)| \leq 2$. Thus, for any two intra-level edges $\{u, v\}$ and $\{u', v'\}$ we have $||\mathrm{bfs}(v) - \mathrm{bfs}(u)| - |\mathrm{bfs}(v') - \mathrm{bfs}(u')|| \leq 1$, i. e., the bandwidth of the BFS order is 1 and we are done [30].

The graph from Fig. 11 has queue number three. This was checked by exhaustive search using a SAT solver. □

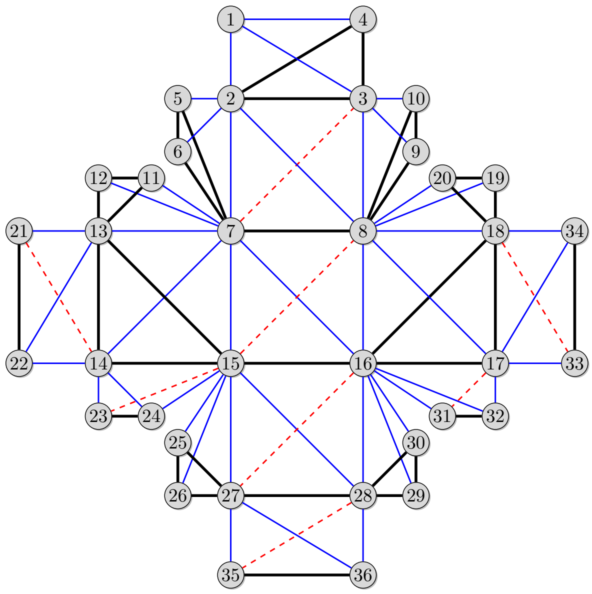


Fig. 11: Example for the proof of Theorem 7 that every *o1p* graph has a 3-queue-layout. The vertices are labeled by their BFS number. Inter-level edges are drawn dashed red and solid blue. Intra-level edges are drawn thick black. The depicted graph has no 2-queue-layout.

We do not know how to efficiently compute the queue number of an *o1p* graph. So the complexity of this problem remains open, although the output is $1,2$ or 3.

Another prominent graph parameter is the *chromatic number*, $\chi(G)$, which is the minimum number of colors for the vertices such that adjacent vertices have different colors. $\chi(G)=2$ if and only if G is bipartite and this can be tested in linear time. The chromatic number of a planar graph is at most four [2,3], but its computation is NP-hard [28]. Also *o1p* graphs may need four colors for their K_4s. However, for *o1p* graphs the chromatic number can be computed in linear time since *o1p* graphs have bounded treewidth [4].

Corollary 11 *Every o1p graph is 4-colorable and there is a linear time algorithm to compute the chromatic number of an o1p graph.*

5 Drawings

We consider three types of drawings, straight-line, visibility representations, and in 3D. Eggleton [24] proved the existence of a straight-line and convex drawing for maximal *o1p* graphs. He showed that edges must cross in the interior of a convex quadrangle. However, he neither provided a bound on the area nor an efficient algorithm. Both are readily obtained from the fact that *o1p* graphs are planar, and can thus be drawn straight-line in quadratic area [27, 39]. Can we do better? Can we draw outerplanar?

For the subclass of outerplanar graphs Di Battista and Frati [17] showed that they can be drawn straight-line in $\mathcal{O}(n^{1.48})$ area and Biedl [9] established a visibility representation in $\mathcal{O}(n\log n)$ area. Dehkordi and Eades [15] proved that every *o1p* drawing can be transformed into a right angle crossing drawing with all vertices in the outer face and right angle crossings, preserving the embedding but at the expense of exponential area. For the larger class of 1-planar graphs, there are straight-line drawings in exponential area if there are no B- or W-configurations [32, 42], and Alam et al. [1] showed that every 3-connected 1-planar graph has a straight-line drawing on a grid of quadratic size (with the exception of a single edge in the outer face).

A visibility representation draws the vertices as horizontal lines between grid points, such that these lines do not overlap. Two such lines must see each other along a vertical line if there is an edge between the displayed vertices. It is a well-known fact that every planar graph has a visibility representation in $\mathscr{O}(n^2)$ area, which can be computed in linear time, see also [16]. Here, we use Biedl's algorithm for compact visibility representations.

Theorem 8 *Every o1p graph has a planar visibility representation in $\mathscr{O}(n \log n)$ area, and the representation can be computed in linear time.*

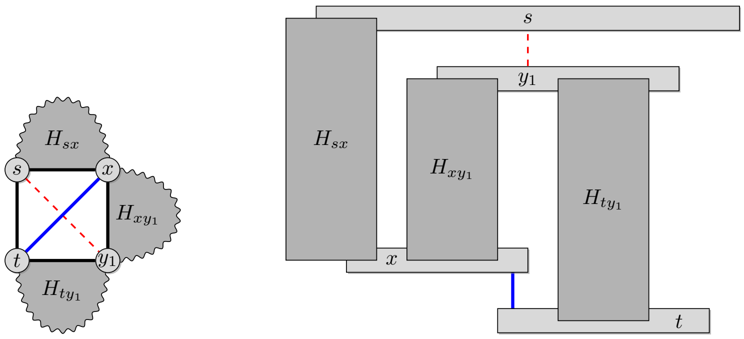
Proof First, augment the given graph G to a (plane-) maximal *o1p* graph as described in Sect. 3 and choose an *o1p* embedding. There is a K_4 induced by the vertices of each pair of crossing edges by Proposition 1. In accordance with Biedl's algorithm [9] we call the vertices s,t,x,y_1 and let the edges $\{s,y_1\}$ and $\{t,x\}$ cross. For each such K_4 remove $\{s,y_1\}$. The remaining graph is maximal outerplanar and Biedl's algorithm inductively computes a visibility representation with the following property. For a triangle (s,t,x) with the edge $\{s,t\}$ in the outer face, the bars of s and t are on top and bottom, and the bar of x is just above the bar of t. (Biedl uses an extended flat visibility representation which allows to place the bars of x and t on a horizontal line). If the edge $\{t,x\}$ is removed, the roles of s and t are exchanged.

In each inductive step the edge $\{s,t\}$ is in the outer face of the actual (sub-) graph H. H is composed of three subgraphs H_{sx}, H_{xy_1} and H_{ty_1}, which each consist of the respective edge and the outer face, see Fig. 12. The visibility representations of the subgraphs H_{sx}, H_{xy_1} and H_{ty_1} are placed between the bars of s, x, and t such that those of H_{xy_1} and H_{ty_1} are flipped and the bar of y_1 is just below the bar of s. The flip changes the embedding of the original K_4 and places y_1 in the triangle (s,t,x). By construction, the bars of s and y_1 are adjacent and can see each other, such that a removed edge $\{s,y_1\}$ can be represented without any expansion of the area. Biedl's algorithm uses $\mathscr{O}(n \log n)$ area, and runs in linear time. Also, the (planar) maximal augmentation, the inductive removal of edges in K_4s and the addition of their visibility lines take linear time. For an illustration, see Fig. 13. □

This visibility representation changes the embedding and does not display outerplanarity and crossings. Drawings that respect these criteria can be obtained using the algorithm of Alam et al. [1]:

Theorem 9 *Every o1p graph has a straight-line grid drawing in $\mathscr{O}(n^2)$ area such that all vertices are in the outer face, and the drawing can be computed in linear time.*

Proof First, augment the given *o1p* graph G to a maximal *o1p* graph as described in Sect. 3. Then add a new vertex t in the outer face and connect t with all vertices. The so obtained graph is 3-connected and 1-planar, and can be drawn by using the algorithm from [1]. This algorithm uses a canonical ordering and the shift technique from [27]. Choose two adjacent vertices of G as the two base vertices 1 and 2 and let t be the top vertex. Then the vertices of G are placed as the contour below t from the leftmost vertex 1 to the rightmost vertex 2. Finally, t and its incident edges and the edges added in the augmentation phase are omitted. The correctness and the area bounds follow from [1] and all phases take linear time. For an illustration, see Fig. 14. □



(a) K_4 with subgraphs (b) Schematic visibility representation

Fig. 12: Inductive step of Biedl's algorithm [9], where the red dashed edge is removed but can be represented.

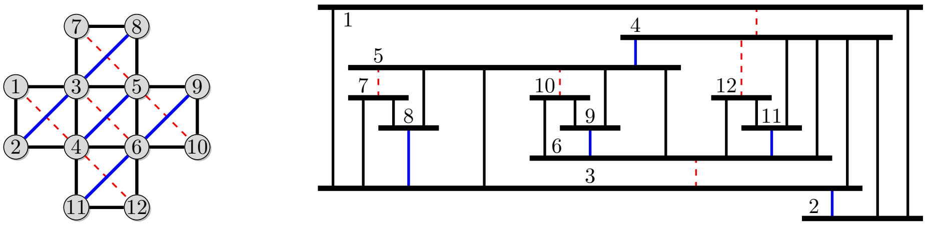
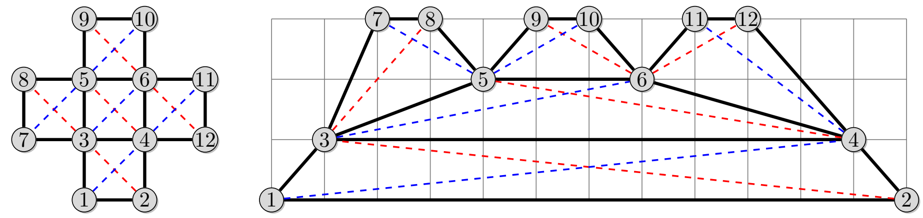


Fig. 13: The "doublecross" graph and its visibility representation.



(a) Input embedding (b) *o1p* drawing

Fig. 14: The "doublecross" graph as example for the embedding preserving drawing algorithm. The vertices are labeled by a canonical ordering. Crossed edges (dashed red and blue) are temporarily removed.

From the bounded treewidth and [21] we obtain:

Corollary 12 *Every o1p graph has a 3-dimensional straight-line drawing in linear volume.*

6 Conclusion and Open Problems

We have designed a linear-time recognition algorithm for *o1p* that in the positive case returns a witness in terms of an *o1p* embedding and in the negative case detects one of six minors. Moreover, we have characterized *o1p* graphs by common properties and measures and have compared the results to planar, 1-planar and outerplanar graphs. This shows that *o1p* graphs are far more manageable than their superclass of 1-planar graphs. However, some problems are still open. For example, outerplanar graphs can be drawn in sub-quadratic area of size $\mathcal{O}(n^{1.48})$ [17]. Can this bound be achieved for *o1p* graphs? Is it possible to compute the queue number of an *o1p* graph efficiently?

Acknowledgements We thank the anonymous referees for their careful reading and useful comments and suggestions.

References

1. Alam, M.J., Brandenburg, F.J., Kobourov, S.G.: Straight-line drawings of 3-connected 1-planar graphs. In: S. Wismath, A. Wolff (eds.) Graph Grawing, GD 2013, *Lecture Notes in Computer Science*, vol. 8242, pp. 83–94. Springer (2014)
2. Appel, K., Haken, W.: Every planar map is four colorable. Part I. discharging. Illinois Journal of Mathematics **21**, 429–490 (1977)
3. Appel, K., Haken, W., Koch, J.: Every planar map is four colorable. Part II. reducibility. Illinois Journal of Mathematics **21**, 491–567 (1977)
4. Arnborg, S., Proskurowski, A.: Linear time algorithms for NP-hard problems restricted to partial *k*-trees. Discrete Applied Mathematics **23**(1), 11–24 (1989)
5. Auer, C., Bachmaier, C., Brandenburg, F.J., Gleißner, A., Hanauer, K., Neuwirth, D., Josef, R.: Recognizing outer 1-planar graphs in linear time. In: S. Wismath, A. Wolff (eds.) Graph Grawing, GD 2013, *Lecture Notes in Computer Science*, vol. 8242, pp. 107–118. Springer (2014)
6. Auer, C., Brandenburg, F.J., Gleißner, A., Reislhuber, J.: 1-planarity of graphs with a rotation system. Journal of Graph Algorithms and Applications **19**(1), 67–86 (2015)
7. Bannister, M.J., Cabello, S., Eppstein, D.: Parameterized complexity of 1-planarity. In: F. Dehne, R. Solis-Oba, J.R. Sack (eds.) WADS 2013, pp. 97–108 (2013)
8. Bernhart, F., Kainen, P.C.: The book thickness of a graph. Journal of Combinatorial Theory, Series B **27**(3), 320–331 (1979)
9. Biedl, T.C.: Small drawings of outerplanar graphs, series-parallel graphs, and other planar graphs. Discrete & Computational Geometry **45**(1), 141–160 (2011)
10. Bodendiek, R., Schumacher, H., Wagner, K.: Bemerkungen zu einem Sechsfarbenproblem von G. Ringel. Abh. aus dem Math. Seminar der Univ. Hamburg **53**, 41–52 (1983)
11. Bodlaender, H.L.: A linear-time algorithms for finding tree-decompositions of small treewidth. SIAM Journal on Computing **25**(6), 1305–1317 (1996)
12. Brandenburg, F.J., Eppstein, D., Gleißner, A., Goodrich, M.T., Hanauer, K., Reislhuber, J.: On the density of maximal 1-planar graphs. In: W. Didimo, M. Patrignani (eds.) Graph Grawing, GD 2012, *Lecture Notes in Computer Science*, vol. 7704, pp. 327–338. Springer (2013)
13. Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph Classes. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia (1999)
14. Cabello, S., Mohar, B.: Adding one edge to planar graphs makes crossing number and 1-planarity hard. SIAM Journal on Computing **42**(5), 1803–1829 (2013)
15. Dehkordi, H.R., Eades, P.: Every outer-1-plane graph has a right angle crossing drawing. Journal of Computational Geometry and Applications **22**(6), 543–558 (2012)
16. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice Hall (1999)
17. Di Battista, G., Frati, F.: Small area drawings of outerplanar graphs. Algorithmica **54**(1), 25–53 (2009)
18. Di Battista, G., Frati, F., Pach, J.: On the queue number of planar graphs. SIAM Journal on Computing **42**(6), 2243–2285 (2013)
19. Di Battista, G., Tamassia, R.: On-line planarity testing. SIAM Journal on Computing **25**(5), 956–997 (1996)
20. Didimo, W.: Density of straight-line 1-planar graph drawings. Information Processing Letters **113**(7), 236–240 (2013)
21. Dujmović, V., Morin, P., Wood, D.R.: Layout of graphs with bounded tree-width. SIAM Journal on Computing **34**(3), 553–579 (2005)
22. Eades, P., Hong, S.H., Katoh, N., Liotta, G., Schweitzer, P., Suzuki, Y.: Testing maximal 1-planarity of graphs with a rotation system in linear time. Theoretical Computer Science **513**, 65–76 (2013)
23. Eades, P., Liotta, G.: Right angle crossing graphs and 1-planarity. Discrete Applied Mathematics **161**(7–8), 961–969 (2013)

24. Eggleton, R.B.: Rectilinear drawings of graphs. Utilitas Mathematica **29**, 149–172 (1986)
25. Fabrici, I., Madaras, T.: The structure of 1-planar graphs. Discrete Mathematics **307**(7–8), 854–865 (2007)
26. Frati, F.: Straight-line drawings of outerplanar graphs in $\mathcal{O}(dn\log n)$ area. Computational Geometry **45**(9), 524–533 (2012)
27. de Fraysseix, H., Pach, J., Pollack, R.: How to draw a planar graph on a grid. Combinatorica **10**, 41–51 (1990)
28. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. Freeman (1979)
29. Gutwenger, C., Mutzel, P.: A linear time implementation of SPQR-trees. In: J. Marks (ed.) Graph Drawing, GD 2000, *Lecture Notes in Computer Science*, vol. 1984, pp. 77–90. Springer (2001)
30. Heath, L.S., Rosenberg, A.L.: Laying out graphs using queues. SIAM Journal on Computing **21**(5), 927–958 (1992)
31. Hliněný, P.: Crossing number is hard for cubic graphs. Journal of Combinatorial Theory, Series B **96**(4), 455–471 (2006)
32. Hong, S.H., Eades, P., Liotta, G., Poon, S.H.: Fáry's theorem for 1-planar graphs. In: J. Gudmundsson, J. Mestre, T. Viglas (eds.) Computing and Combinatorics Conference, COCOON 2012, *Lecture Notes in Computer Science*, vol. 7434, pp. 335–346. Springer (2012)
33. Hong, S.H., Eades, P., Naoki, K., Liotta, G., Schweitzer, P., Suzuki, Y.: A linear-time algorithm for testing outer-1-planarity. Algorithmica (2014). Published online
34. Korzhik, V.P., Mohar, B.: Minimal obstructions for 1-immersion and hardness of 1-planarity testing. Journal of Graph Theory **72**, 30–71 (2013)
35. Mac Lane, S.: A structural characterization of planar combinatorial graphs. Duke Mathematical Journal **3**(3), 460–472 (1937)
36. Mitchell, S.L.: Linear algorithms to recognize outerplanar and maximal outerplanar graphs. Information Processing Letters **9**(5), 229–232 (1979)
37. Pach, J., Tóth, G.: Graphs drawn with a few crossings per edge. Combinatorica **17**, 427–439 (1997)
38. Ringel, G.: Ein Sechsfarbenproblem auf der Kugel. Abh. aus dem Math. Seminar der Univ. Hamburg **29**, 107–117 (1965)
39. Schnyder, W.: Embedding planar graphs on the grid. In: ACM-SIAM Symposium on Discrete Algorithms, SODA 1990, pp. 138–147. SIAM (1990)
40. Thomassen, C.: Planarity and duality of finite and infinite graphs. Journal of Combinatorial Theory, Series B **29**, 244–271 (1980)
41. Thomassen, C.: Kuratowski's theorem. Journal of Graph Theory **5**(3), 225–241 (1981)
42. Thomassen, C.: Rectilinear drawings of graphs. Journal of Graph Theory **12**(3), 335–341 (1988)
43. Wigderson, A.: The complexity of the Hamiltonian circuit problem for maximal planar graphs. Tech. Rep. 298, Department of EECS, Princeton University (1982)
44. Williamson, S.G.: Depth-first search and Kuratowski subgraphs. Journal of the ACM **31**(4), 681–693 (1984)