Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 25, no. 1, pp. 481–512 (2021)
DOI: 10.7155/jgaa.00568

On Strict (Outer-)Confluent Graphs

Henry Förster [1] *Robert Ganian* [2] *Fabian Klute* [3] *Martin Nöllenburg* [2]

[1]University of Tübingen, Tübingen, Germany
[2]Algorithms and Complexity Group, TU Wien, Vienna, Austria
[3]Utrecht University, Utrecht, The Netherlands

Submitted: December 2020	Reviewed: April 2021	Revised: July 2021
Accepted: August 2021	Final: August 2021	Published: September 2021
Article type: Regular Paper		Communicated by: W. Evans

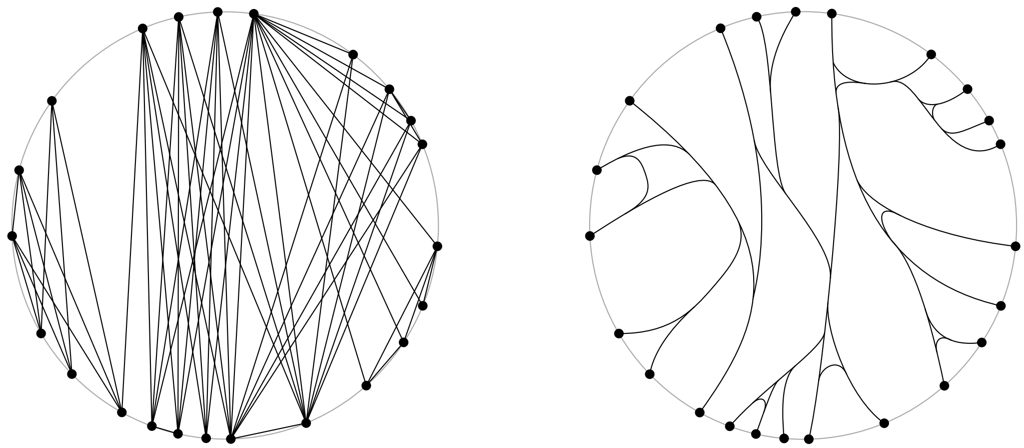
Abstract. A strict confluent (SC) graph drawing is a drawing of a graph with vertices as points in the plane, where vertex adjacencies are represented not by individual curves but rather by unique smooth paths through a planar system of junctions and arcs. If all vertices of the graph lie in the outer face of the drawing, the drawing is called a strict outerconfluent (SOC) drawing. SC and SOC graphs were first considered by Eppstein et al. in Graph Drawing 2013. Here, we establish several new relationships between the class of SC graphs and other graph classes, in particular string graphs and unit-interval graphs. Further, we extend earlier results about special bipartite graph classes to the notion of strict outerconfluency, show that SOC graphs have cop number two, and establish that tree-like (Δ-)SOC graphs have bounded cliquewidth.

1 Introduction

Confluent drawings of graphs are geometric graph representations in the Euclidean plane, in which vertices are mapped to points, but edges are not drawn as individually distinguishable geometric objects. Instead, an edge between two vertices u and v is represented by a smooth path between the points of u and v through a crossing-free system of arcs and junctions. Since multiple edge representations may share some arcs and junctions of the drawing, this allows dense and non-planar graphs to be drawn in a plane way. An example is shown in Figure 1. Hence confluent drawings can be seen as theoretical counterpart of heuristic edge bundling techniques, which are frequently used in network visualizations to reduce visual clutter in layouts of dense graphs [3, 38, 56].

Robert Ganian acknowledges support by the Austrian Science Fund (FWF, projects P31336 and Y1329). A preliminary version of this manuscript appeared in the Proceedings of the International Symposium on Graph Drawing and Network Visualization 2019.

E-mail addresses: foersth@informatik.uni-tuebingen.de (Henry Förster) rganian@ac.tuwien.ac.at (Robert Ganian) f.m.klute@uu.nl (Fabian Klute) noellenburg@ac.tuwien.ac.at (Martin Nöllenburg)



(a) A rather complex circular layout. (b) The clearer strict outerconfluent diagram.

Figure 1: A straight line circular drawing and a strict outerconfluent diagram of the same graph. The gray circle is only added for illustration purposes and does not represent edges.

More formally, a *confluent drawing* D of a graph $G = (V, E)$ consists of a set of points representing the vertices of G, a set of junction points, and a set of smooth arcs, such that each arc starts and ends at either a vertex point or a junction, no two arcs intersect (except at common endpoints), and all arcs meeting in a junction share the same tangent line in the junction point. There is an edge $uv \in E$ if and only if there is a smooth path from u to v in D not passing through any other vertex.

Eppstein et al. [22] defined the class of strict confluent (SC) drawings, which require that every edge of the graph must be represented by a unique smooth path and that there are no self-loops. They showed that for general graphs it is NP-complete to decide whether an SC drawing exists. An SC drawing is called *strict outerconfluent* (SOC) if all vertices lie on the boundary of a (topological) disk that contains the SC drawing. For graphs with a given cyclic vertex order, Eppstein et al. [22] presented a constructive efficient algorithm for testing the existence of an SOC drawing. Without a given vertex order, neither the recognition complexity nor a characterization of such graphs is known. We approach the characterization problem by comparing the SOC graph class with a hierarchy of classes of geometric intersection graphs. Since confluent drawings make heavy use of intersecting curves to represent edges in a planar way, it seems natural to ask what kind of geometric intersection models can represent a confluent graph.

In our first result we show that all S(O)C graphs are in fact (outer-) string graphs [44], giving a first class of geometric intersection graphs that contains all strict (outer-)confluent graphs. We also show that every unit interval graph can be drawn as a strict confluent diagram and that every bipartite permutation graph without dominos (two C_4 glued at an edge) can be drawn as a strict outerconfluent graph. In fact, our drawing keeps the bipartite sets of the graph on two parallel lines and draws the confluent arcs between these lines. This is similar to the technique used by Hui et al. [41]. Following their naming scheme, we call our drawings *strict bipartite-outerconfluent drawings*. We also prove that excluding induced domino graphs is really necessary by showing that there are bipartite permutation graphs that cannot be drawn strict outerconfluent if we allow induced domino graphs. We furthermore show that many natural subclasses of outer-string graphs are incomparable to SOC graphs. More specifically, we show among others that circle [26], circular-arc [40], series-parallel [50], chordal [29], co-chordal [7], and co-comparability [33] graphs are all incomparable to SOC graphs. This list may help future research by excluding a series of natural candidates for sub- and super-classes of SOC graphs. Inspired by earlier work of Gavenčiak et al. [28], we examine the cop-number of SOC graphs and show that it is at most two, suggesting that SOC graphs are in fact more comparable to interval-filament graphs [28, 30]. Finally, our last result shows that the clique-width of so-called tree-like Δ-SOC graphs is bounded by a constant, generalizing a previous result of Eppstein et al. [21]. A graphical overview of the results in this paper is presented in Figure 2.

Related work Confluent drawings were introduced by Dickerson et al. [16], who identified classes of graphs that admit or do not admit confluent drawings. Subsequently, the notions of strong and tree confluency have been introduced [41], as well as Δ-confluency [21]. Confluent drawings have further been used for drawings of layered graphs [20] and Hasse diagrams [23].

As mentioned above confluent drawings can be seen as a theoretical counterpart to heuristic edge bundling techniques. Holten [38] introduced the first algorithm for bundling edges. His approach can be used for graph-data that observes some form of hierarchy between sets of vertices. Later, Holten and van Wijk [39] also introduced force directed bundling, in which the edges are attracted and repelled from each other. This technique is similar to a force directed graph layout computed, for example, by a spring embedder [42]. Other heuristic techniques have been developed since then. A summary of such techniques was presented by Zhou et al. [57]. We also recommend the introduction to Bach et al. [3] for a good overview of general bundling techniques.

Commonly, these techniques have a certain degree of ambiguity when it comes to determining the adjacency relations. Bach et al. [3] proposed to use techniques inspired by confluent graph layouts for practical edge bundling in which edges that cannot be bundled unambiguously are allowed to cross at sharp angles. Recently, Zheng et al. [56] improved Bach et al.'s approach. These recent publications show that confluent edge bundling has potential for practical applications, extending beyond the theoretical interest of researchers in graph drawing. Their techniques make use of so-called *power graphs*. There, the idea is to find groups of vertices that have similar sets of adjacent vertices. The edges between these groups can then be drawn as bundled connections. This technique has been used in the analysis of proteins [49]. A later user study by Dwyer et al. [19] showed that the shortest path task can be solved faster than in regular node-link visualizations. Dwyer et al. [18] later presented an efficient heuristic approach to compute such layouts.

Confluent drawings are not the only theoretical model to conceptualize edge bundling. Recent years have seen several results on *bundled crossings* and the *bundled crossing number* of a graph. Given a graph $G = (V, E)$ the bundled crossing number is defined as the minimum number of bundled crossings over all drawings of G. A bundled crossing is simply a set of crossings between edges $E' \subseteq E$ that is such that the contributing edges in E' can be partitioned into two sets $E'_1, E'_2 \subseteq E'$ with the property that no two edges in E'_1 or E'_2 cross inside a polygonal region around the crossings of edges in E'. Originally this concept was defined by Fink et al. [25] and studied by Alam et al. [2] in the same year. Their results show that computing the bundled crossing number is, unsurprisingly, NP-hard in general and also in embedded graphs. Some constant-factor approximation algorithms are presented as well. Recently, Chaplick et al. [11] proved that for circular drawings the problem is fixed-parameter tractable in the number of bundled crossings. The bundled crossing number is also closely related to *block crossings* in storyline visualizations [52, 53].

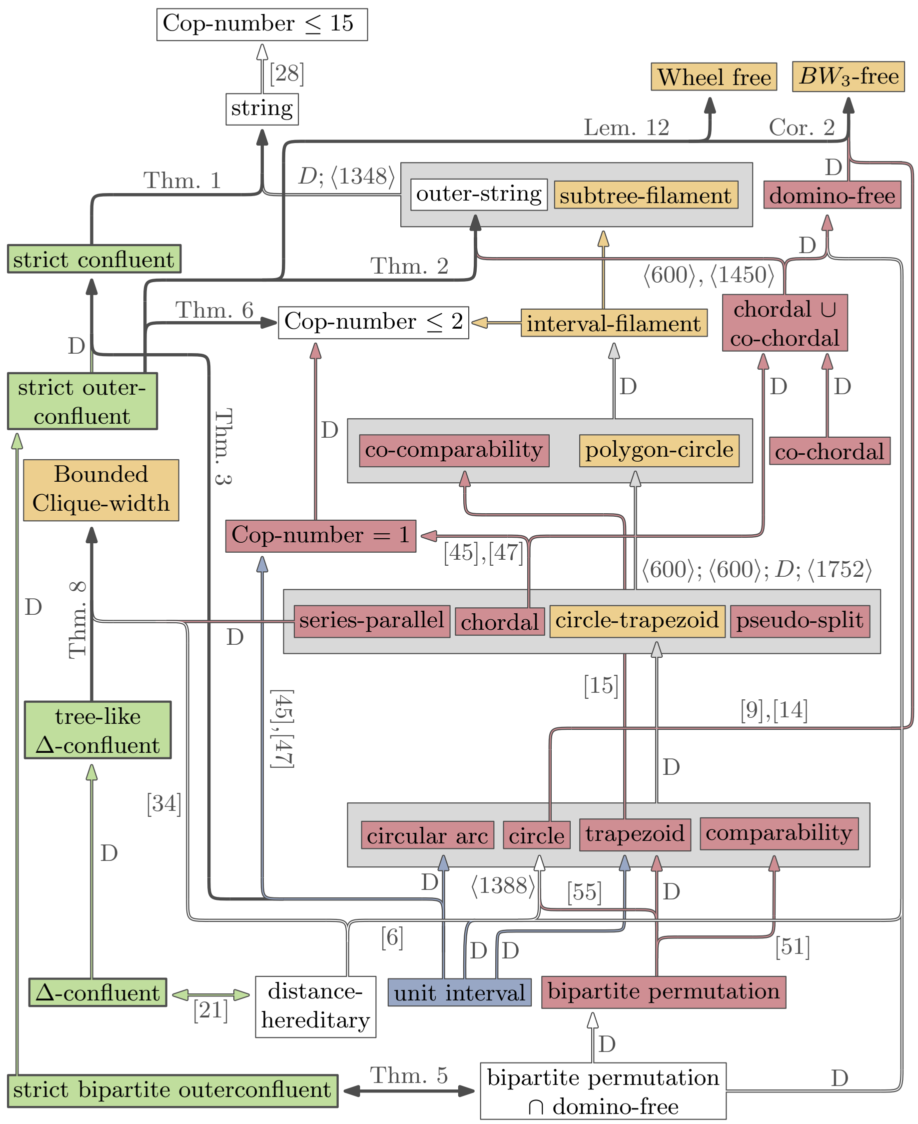


Figure 2: Inclusions among graph classes related to SOC graphs. Arrows point from sub- to superclass, where edge label 'D' marks an inclusion by definition. Fat arrows are inclusions shown in this paper and are labelled with the corresponding theorem. Green boxes are confluent graph classes. Red boxes are classes that are incomparable to SOC graphs. Many of these incompatibilities are new results and proven in Theorem 7. Orange boxes are classes that are potential superclasses of SOC graphs. Formal arguments why these classes are not included in the SOC graphs are given in Corollary 3. Blue boxes are potential subclasses of the SOC graphs. The numbers in $\langle\cdot\rangle$ indicate references of graphclasses.org. Numbers separated by ';' are the references for the corresponding classes in a grey box from left to right.

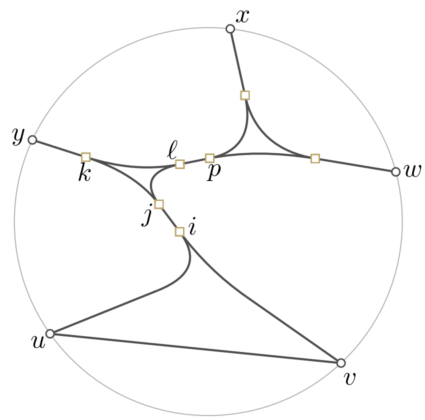


Figure 3: A strict outerconfluent diagram representing K_5. Nodes are disks, junctions are squares.

Our results After introducing basic definitions and properties in Section 2, we show in Section 3 that SC and SOC graphs are, respectively, string and outerstring graphs. Section 4 shows that every unit interval graph [48, 54] can be drawn strict confluent. In Section 5, we consider the strict bipartite-outerconfluent drawings and show that they coincide with the bipartite permutation graphs when excluding induced dominos. We examine the cop-number of SOC graphs in Section 6 and show that it is at most two. Section 7 contains our series of non-inclusion results as well as the counterexample for the existence of strict outerconfluent drawings for general bipartite permutation graphs. In Section 8, we show that the clique-width of so-called tree-like Δ-SOC graphs is bounded by a constant. Finally, we conclude in Section 9 and discuss open problems and future research directions with respect to strict (outer-)confluent graphs.

2 Preliminaries

A *confluent diagram* $D = (N, J, \Gamma)$ in the plane $\mathbb{R}^2$ consists of a set N of points called *nodes*, a set J of points called *junctions* and a set Γ of simple smooth curves called *arcs* whose endpoints are in $J \cup N$. Further, two arcs may only intersect at common endpoints. If they intersect in a junction they must share the same tangent line. We illustrate this in Figure 3.

Let $D = (N, J, \Gamma)$ be a confluent diagram and let $u, v \in N$ be two nodes. A *uv-path* $p = (\gamma_0, \ldots, \gamma_k)$ in D is a sequence of arcs $\gamma_0 = (u, j_1), \gamma_1 = (j_1, j_2), \ldots, \gamma_k = (j_k, v) \in \Gamma$ such that $j_1, \ldots j_k$ are junctions and p is a smooth curve. In Figure 3 the unique uy-path passes through junctions i, j, k. If there is at most one uv-path for each pair of nodes u, v in N and if there are no self-loops, i.e., no uu-path for any $u \in N$, we say that D is a *strict* confluent diagram. The uniqueness of uv-paths and the absence of self-loops imply that every uv-path is actually a path in the graph-theoretic sense, where no vertex is visited twice. We further define $P(D)$ as the set of all smooth paths between all pairs of nodes in N. Let $p \in P(D)$ be a path and $j \in J$ a junction in D, then we write $j \in p$, if p passes through j.

As observed by Eppstein et al. [22], we may assume that every junction is a *binary* junction, where exactly three arcs meet such that the three enclosed angles are $180°, 180°, 0°$. In other words two arcs from the same direction merge into the third arc, or, conversely, one arc splits into two arcs. A (strict) confluent diagram with higher-degree junctions can easily be transformed into an equivalent (strict) one with only binary junctions.

Let $j \in J$ be a binary junction with the three incident arcs $\gamma_1, \gamma_2, \gamma_3$. Let the angle enclosed by γ_1 and γ_2 be 0° and the angle enclosed by γ_3 and γ_1 (or γ_2) be 180°. Then we say that j is a *merge-junction* for γ_1 and γ_2 and a *split-junction* for γ_3. We also say that γ_1 and γ_2 *merge* at j and that γ_3 *splits* at j. Given two nodes $u, v \in N$ and a junction $j \in J$ we say that j is a merge-junction for u and v if there is a third node $w \in N$, a uw-path p and a vw-path q such that $j \in p$ and $j \in q$, the respective incoming arcs $\gamma_p = (j_p, j)$ and $\gamma_q = (j_q, j)$ are distinct and the suffix paths of p and q from j to w are equal. Conversely, we say that a junction $j \in J$ is a split-junction for a node $u \in N$ if there are two nodes $v, w \in N$, a uv-path p, and a uw-path q such that $j \in p$ and $j \in q$, the prefix paths of p and q from u to j are equal and the respective subsequent arcs $\gamma_p = (j, j_p)$ and $\gamma_q = (j, j_q)$ are distinct. In Figure 3, junction i is a merge-junction for u and v, while it is a split junction for each of w, x, y. Two junctions $i, j \in J$ are called a *merge-split pair* if i and j are connected by an arc γ and both i and j are split-junctions for γ; in Figure 3, junctions i and j form a merge-split pair, as well as junctions ℓ and p.

We call an arc $\gamma \in \Gamma$ *essential* if we cannot delete γ without changing adjacencies in the represented graph. We call a confluent diagram D *reduced*, if every arc is essential. Notice that this is a different notion than strictness, since it is possible that in a reduced confluent diagram we find two different paths between a pair of nodes. Without loss of generality we can assume that the nodes of an outerconfluent diagram are placed on a circle with all arcs and junctions inside the circle. We can infer a *cyclic order* π from an outerconfluent diagram D by walking clockwise around the boundary of the unbounded face and adding the nodes to π in the order they are visited.

From a confluent diagram $D = (N, J, \Gamma)$ we derive a simple, undirected graph $G_D = (V_D, E_D)$ with $V_D = N$ and $E_D = \{uv \mid \exists\, uv\text{-path } p \in P(D)\}$. We say D is a confluent drawing of a graph G if G is isomorphic to G_D and that G is a (strict) (outer-)confluent graph if it admits a (strict) (outer-)confluent drawing.

3 Strict (Outer-)Confluent ⊂ (Outer-)String

The class of *string graphs* [44] contains all graphs $G = (V, E)$ which can be represented as the intersection graph of open curves in the plane. We show that they form a superclass of SC graphs and that every SOC graph is an outer-string graph [44]. *Outer-string* graphs are string graphs that can be represented so that strings lie inside a disk and intersect the boundary of the disk in one endpoint. Note that strings are allowed to self-intersect and cross each other more than once.

Let $D = (N, J, \Gamma)$ be a strict confluent diagram. For every node $u \in N$ we construct the *junction tree* T_u of u, with root u and a leaf for each neighbor v of u in G_D. The interior vertices of T_u are the junctions which lie on the (unique) uv-paths. The strictness of D implies that T_u is a tree. Observe that every internal node of T_u has at most two children. Further, every merge-junction for u is a vertex with one child in T_u, and every split-junction for u has two children. For every junction j in T_u we can define the subtree $T_{u,j}$ of T_u with root j.

Lemma 1 *Let $D = (N, J, \Gamma)$ be a strict confluent diagram, let $u, v \in N$ be two nodes and let i, j be two distinct merge-junctions for u, v. Then i is neither an ancestor nor a descendant of j in T_u (and, by symmetry, in T_v).*

Proof: Assume i, j would be two such junctions, where i is an ancestor of j in T_u. Then there are two distinct smooth paths from v to j in D: one passing through i and then following the path to j in T_u, the other one merging into T_u in junction j. This contradicts the strictness of D. □

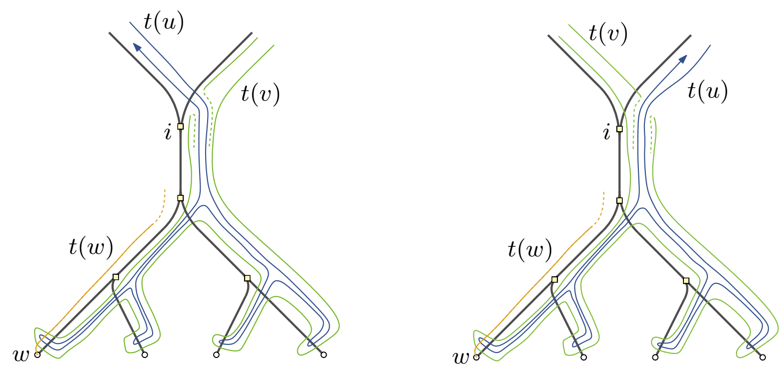


Figure 4: Two possible configurations for inserting a new trace $t(u)$ that meets an existing trace $t(v)$ at a merge junction i; $t(v)$ is cut and re-routed.

To create a string representation of an SC graph we trace the paths of a strict confluent diagram $D = (N, J, \Gamma)$, starting from each node $u \in N$ and combine them into a string representation. Figure 4 shows an example. We traverse the junction tree for each $u \in N$ on the left-hand side of each arc (seen from its root u) and create a string $t(u)$, the *trace* of u, with respect to T_u as follows.

Start from u and traverse T_u in left-first DFS order. Upon reaching a leaf ℓ make a clockwise U-turn and backtrack along the left-hand side of the arc (as seen from the root) to the previous split-junction of T_u. When returning to a split-junction we have two cases. (a) coming from the left subtree: cross the arc from the left subtree at the junction and descend into the right subtree. (b) coming from the right subtree: cross the arc to the left subtree again and backtrack upward in the tree along the existing trace to the previous split-junction of T_u.

Finally, at a merge-junction i with at least one trace from the other arc merging into i already drawn: Let $v \in N$ be such that u and v merge at i and $t(v)$ has already traced the subtree $T_{u,i} = T_{v,i}$. In this case we temporarily cut open the part of trace $t(v)$ closest to $t(u)$, route $t(u)$ through the gap and let it follow $t(v)$ along $T_{u,i}$ until it returns to junction i, where $t(u)$ passes through the gap again. Since $T_{u,i} = T_{v,i}$ this is possible without $t(u)$ intersecting $t(v)$. Now it remains to reconnect the two open ends of $t(v)$, but this can again be done without any new intersections by winding $t(v)$ along the "outside" of $t(u)$. See Figure 4 for an illustration. If there are more than two traces merging at i, they can all be treated as a single "bundled" trace within $T_{u,i}$.

Theorem 1 *Every SC graph is a string graph.*

Proof: Given an SC graph $G = (V, E)$ with a strict confluent drawing $D = (N, J, \Gamma)$ we construct the traces as described above for every node $u \in N$. In the following let u, v be two nodes of D. We distinguish three cases.

Case 1 (uv-path in $P(D)$): We draw $t(u)$ and $t(v)$ as described above. Since there is a uv-path in $P(D)$ we have to guarantee that $t(u)$ and $t(v)$ intersect at least once. We introduce crossings at the leaves corresponding to u and v in T_v and T_u when $t(u)$ and $t(v)$ make a U-turn; see how the trace $t(u)$ intersects $t(w)$ near the leaf w in Figure 4. Since we only need to guarantee at least one crossing, we can introduce the crossings at both leaves without problem. Moreover, note that it is not an issue that routing the traces like this forces other traces to also make a crossing at the leaf, since a trace is only present at a leaf if the corresponding path exists in $P(D)$. See for example trace $t(v)$ in Figure 4 which intersects $t(w)$ at w.

Case 2 (No uv-path in $P(D)$ and u, v share no merge-junction): In this case T_u and T_v are disjoint trees. Traces can meet only at shared junctions and around leaves, but since $t(u)$ and $t(v)$ trace disjoint trees, intersections are impossible.

Case 3 (No uv-path in $P(D)$ and u, v share a merge-junction): First assume u and v share a single merge-junction $i \in J$ and assume $t(v)$ is already drawn when creating trace $t(u)$. We have to be careful that $t(v)$ and $t(u)$ do not intersect. If we route the traces at the merge-junction i as depicted in Figure 4, they visit the shared subtree $T_{u,i} = T_{v,i}$ without intersecting each other.

Now assume u and v share $k > 1$ merge-junctions $j_1, \ldots, j_k \in J$ and u and v merge at each j_i. Consequently, we find k shared subtrees $T^1, \ldots, T^k$. By Lemma 1, however, we know that the intersection of these subtrees is empty. Hence, we can treat every merge-junction and its subtree independently as in the case of a single merge-junction.

These are all the cases for how two junction trees can interact. Hence, the traces $t(u)$ and $t(v)$ for nodes $u, v \in N$ intersect if and only if there is a uv-path in $P(D)$ and, equivalently, the edge $uv \in E_D$. Further, every trace is a continuous curve, so this set of traces yields a string representation of G. □

A construction following the same principle can in fact be used to show:

Theorem 2 *Every SOC graph is an outer-string graph.*

Proof: Let $G = (V, E)$ be an SOC graph with a strict outerconfluent drawing $D = (N, J, \Gamma)$. Construct traces exactly as in the proof of Theorem 1 for the SC graphs. Since for each node $u \in N$ the trace $t(u)$ starts at the position of u in D we immediately know that every trace starts on the boundary of the enclosing disk of D. Further, every trace is a continuous curve and does not leave the enclosing disk of D by construction. Hence, the constructed set of traces immediately yields an outer-string representation of G. □

4 Unit Interval Graphs and Strict Confluent Diagrams

In this section we consider so-called unit interval graphs. Let $G = (V, E)$ be a graph, then G is a unit interval graph if there exists a *unit-interval* layout Γ_{UI} of G, i.e. a representation of G where each vertex $v \in V$ is represented as an interval of unit length and edges are given by the intersections of the intervals.

Theorem 3 *Every unit-interval graph is an SC graph.*

Proof: Consider a unit interval graph G with a unit-interval layout Γ_{UI} of G. In Γ_{UI}, every vertex v is represented by an interval $[\ell(v), r(v)]$ such that $r(v) - \ell(v) = u$ for some constant u, and two vertices v, w are connected by an edge, if and only if $\ell(w) \in [\ell(v), r(v)]$ or $r(w) \in [\ell(v), r(v)]$. We can assume that all intervals have distinct endpoints, as otherwise the following modification is possible: Let $v_1, \ldots, v_k$ be k vertices such that $\ell(v) := \ell(v_1) = \ldots = \ell(v_k)$ and let $\varepsilon > 0$. Then we assign new interval coordinates, such that $\ell(v_i) := \ell(v) + i \cdot \varepsilon$. Clearly, since all vertices $v_1, \ldots, v_k$ had the same neighborhood before, we can chose ε sufficiently small to retain the incidencies. Let $O = (v_1, \ldots, v_n)$ denote the ordering of V such that for vertex v_i it holds that $\ell(v_i) < \ell(v_j)$ for all $j > i$. We call such an ordering O a *left-to-right-ordering.*

We first indentify subcliques $C_1, \ldots, C_k$ of G such that each vertex is part of exactly one C_i as follows. We say that a vertex v_i is a *leader* in a left-to-right-ordering O if and only if there

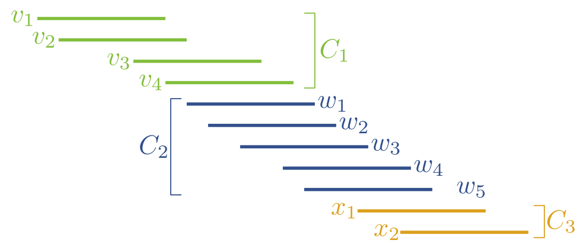


Figure 5: A unit interval graph G with a decomposition of its vertices into a set of cliques as described in the proof of Theorem 3.

exists no vertex v_j with $j < i$ and $\ell(v_i) < r(v_j)$ such that v_j is leader. Note that by definition v_1 is always a leader. The second leader is the first vertex v_i such that $r(v_1) < \ell(v_i)$ and so on. Let $L = (l_1, \dots, l_k) \subseteq V$ denote the left-to-right-ordered set of leaders. It is easy to see that L is a set of disjoint intervals. We say that v_j is *leader of vertex* v_i or $v_j = \text{lead}(v_i)$ if $j \leq i$, v_j is a leader and there exists no leader v_h for $j < h < i$. Observe that every vertex v has a uniquely defined leader which is the interval, in which $\ell(v)$ is located. Since all intervals have unit size, there always exists an edge between vertices with the same leader. Hence, all vertices with the same leader form a clique and we define $C_i = \{v \in V | \text{lead}(v) = l_i\}$. Further, if l_i is leader of vertex v, since all intervals are unit and leaders are disjoint, $r(v) \in (r(l_i), r(l_{i+1}))$, that is, vertex v can only be connected to two leaders. For an illustration of such a decomposition refer to Figure 5.

Next, we describe how to produce a strict confluent diagram D of G. For an example illustration that follows the notation of the proof, refer to Figure 6. Let $\mathcal{C} = (C_1, \dots, C_k)$ denote the left-to-right-ordered (according to their leaders) set of cliques. We draw each clique $C_i = (V_i, V_i \times V_i) \in \mathcal{C}$ with the following SOC layout: Let $V_i = (v_1, \dots, v_k)$ be the left-to-right-ordered set of vertices. We position $v_1, d_2, \dots, d_{k-1}, v_k$ from left to right on a horizontal line H, and we position v_i below d_i for $2 \leq i \leq k-1$, where d_i is a junction connecting v_i with the two neighbors of d_i on H. Note that each of these junctions can be drawn such that they smoothly link each pair of the three incident arcs. We order the drawings of all $C_i \in \mathcal{C}$ from left to right along H, that is, the drawing of C_i appears between the drawings of C_{i-1} and C_{i+1}. Note that all vertices can be reached from below.

It remains to describe how to realize edges from vertices in $C_i \in \mathcal{C}$ to vertices in C_{i+1}. Let $C_i = (V_i, V_i \times V_i)$ and let $V_i = (v_1, \dots, v_k)$ be the left-to-right-ordered set of vertices of C_i. Consider edge $v_\ell w \in E$ for $v_\ell \in V_i$ and $w \in V_{i+1}$. Since $v_\ell w$ exists, in Γ_{UI}, it holds that $\ell(v_\ell) < \ell(w) < r(v_\ell)$. For all $v_j \in V_i$ with $j \geq \ell$, it obviously holds that $\ell(v_j) \in (\ell(v_\ell), \ell(w))$. Therefore, also $(v_j, w) \in E$. Further, for v_ℓ it clearly holds that its neighbors $W \in V_{i+1}$ are consecutive in the left-to-right-order of vertices defined by Γ_{UI}. This allows us to realize the bundle of edges going from V_i to a consecutive set of vertices $W \in V_{i+1}$ as follows. Let $W \subseteq V_{i+1}$ such that v_ℓ is the leftmost neighbor of each $w \in W$. We add a binary junction b_ℓ in between d_ℓ and $d_{\ell-1}$ (or v_1 if $\ell = 2$) such that b_ℓ is a split junction for d_ℓ and a merge junction for $d_{\ell-1}$ and the root b_r of a tree T_b of binary junctions. In T_b, each junction is a split junction for its ancestor and each leaf of T_b is connected to a pair of vertices in W. We position T_b below W and route the segment between b_r and b_ℓ first above the drawing of C_i, then let it cross line H and finally route it below the drawing of C_{i+1}. Since we use this scheme for all C_i, we avoid intersections of segments between different pairs of consecutive cliques. Also, since we directly connect to vertices W via T_b, we realize all edges exactly once, yielding a strict drawing of G. □

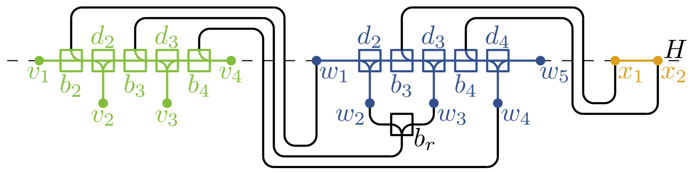


Figure 6: A strict confluent layout of G computed by the algorithm described in the proof of Theorem 3.

5 Strict Bipartite-Outerconfluent Drawings

Let G be a bipartite graph with bipartite sets (X, Y). An outerconfluent drawing of G is *bipartite-outerconfluent* if the vertices in X (and hence also Y) occur consecutively on the boundary. Graphs admitting such a drawing are called *bipartite-outerconfluent*. The *bipartite permutation* graphs are just the graphs that are bipartite and *permutation* graphs, where a permutation graph is a graph that has an intersection model of straight lines between two parallel lines [46].

Theorem 4 (Hui et al. [41]) *The class of bipartite-permutation graphs is equal to the class of bipartite-outerconfluent graphs.*

It is natural to consider the idea of bipartite drawings also in the strict outerconfluent setting. We call a strict outerconfluent drawing D of G *bipartite* if it is bipartite-outerconfluent and strict. The graphs admitting such a drawing are called *strict bipartite-outerconfluent graphs*. In this section we extend Theorem 4 to the notion of strictness.

Given a graph $G = (V, E)$ and a circular ordering of the vertices, we say a crossing between edges $uv, wx \in E$ in the resulting straight line circular layout is *representable* if $G[\{u, v, w, x\}]$ has a $K_{2,2}$ subgraph. We are also going to use the *domino* graph extensively which is just the graph resulting from gluing two 4-cycles together at an edge.

The next lemmas are required in the proof of our theorem.

Lemma 2 *Let D be a strict outerconfluent diagram, G_D the graph represented by D, and π the order inferred from D. If we use π to create a circular straight-line layout of G_D, then every crossing is representable.*

Proof: Let $ac, bd \in E_D$ be two edges that have a crossing in a circular layout of G_D with order π. Let $p, q \in P(D)$ be two paths corresponding to these edges in D. Since the order of the vertices and nodes is the same we know there exist two junctions $i, j \in J$ such that either a, b merge at i and c, d at j or a, d merge at i and c, b at j. This means that the edges (a, b) and (c, d) or (a, d) and (b, c) exist as well, so for every crossing we find that it is part of a $K_{2,2}$. □

It is clear that a graph can only have a strict outerconfluent drawing if it has a circular layout with all crossings representable. This is not sufficient though, as there are such graphs that have no strict outerconfluent drawing. We obtain two 6-vertex obstructions for strict outerconfluent drawings, namely a $K_{3,3}$ with an alternating vertex order, illustrated in Figure 7, and a domino graph in bipartite order, illustrated in Figure 8.

Lemma 3 *Let $G = (V, E)$ be a graph isomorphic to $K_{3,3}$, $X \cup Y = V$ the corresponding bipartite sets, and π the cyclic order of V in which vertices from X and Y alternate. Then there is no strict outerconfluent diagram $D = (N, J, \Gamma)$ with order π and $G_D = G$.*

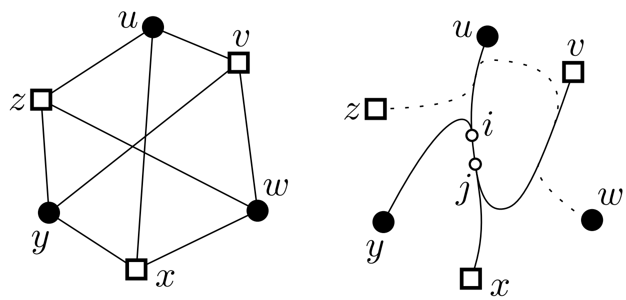


Figure 7: Forbidden alternating order of $K_{3,3}$.

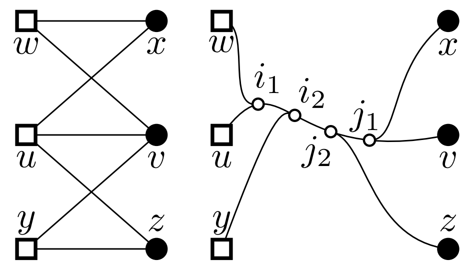


Figure 8: Forbidden domino order.

Proof: Let $G = (V, E)$ and π as above. Draw G in a circular layout with the edges as straight lines. We then find the vertices and edges drawn as in Figure 7.

For contradiction assume that there exists a strict outerconfluent diagram D of G and that no series of arcs in D forms a closed simple curve. We illustrate the argument in Figure 7. Since D is strict we find exactly one path in D for each edge of G. Let $ux \in E$ and $vy \in E$ be two edges that intersect in the above straight-line drawing as showin in Figure 17. Then, the ux-path and vy-path intersect in D and consequently there has to exist a merge-junction $i \in J$ and a split-junction $j \in J$ of u such that y and u merge at i. Conversely, the nodes v and x merge at j. Moreover, there exists a wz-path in D that has to intersect the uv-path or the xy-path in D. Without loss of generality assume it intersects the uv-path. Then, starting from i we can follow the uv-path, then the wz-path, and finally again the uv-path to return to i, a contradiction to the assumption that no simple closed loop exists in D. Moreover, none of the arcs on this loop can be removed without either disconnecting u and v or w and z. □

Lemma 4 *Let $G = (V, E)$ be a graph isomorphic to the domino graph, $X \cup Y = V$ be the corresponding bipartite sets, and π the cyclic order of V in which the vertices in X and in Y are contiguous, respectively. Then there is no strict outerconfluent diagram $D = (N, J, \Gamma)$ with order π and $G_D = G$.*

Proof: The situation and names of the vertices are shown in Figure 8. Each of the two crossings has to be replaced by a series of confluent arcs and junctions. Assume this could be done without two distinct uv-paths existing in the final strict outerconfluent diagram D. Since there need to be paths from u and w to x and v and from u and y to v and z, there exists a merge-split pair $i_1, j_1 \in J$ on the uv-path such that w and u merge at i_1 and v and x merge at j_1. Symmetrically, there exists another merge-split pair $i_2, j_2 \in J$ on the uv-path such that u and y merge at i_2 and v and z merge at j_2. Since either i_1 is before i_2 and j_2 or i_2 is before i_1 and j_1 we find that either there exists an wz- or an yx-path in D, but neither $wz \in E$ nor $yx \in E$.

□

Given that a strict outerconfluent drawing implies a strict outerconfluent drawing for each induced subgraph we find that no strict outerconfluent drawing can contain an induced domino graph or $K_{3,3}$ and its vertices ordered as described in Lemmas 3 and 4, respectively.

Lemma 5 *Suppose that a reduced confluent diagram $D = (N, J, \Gamma)$ with $|N| \geq 6$ contains two distinct uv-paths. Then we can find in $G_D = (V_D, E_D)$ a set $V' \subseteq V_D$ such that $G[V']$ is isomorphic to C_6 with at least one chord.*

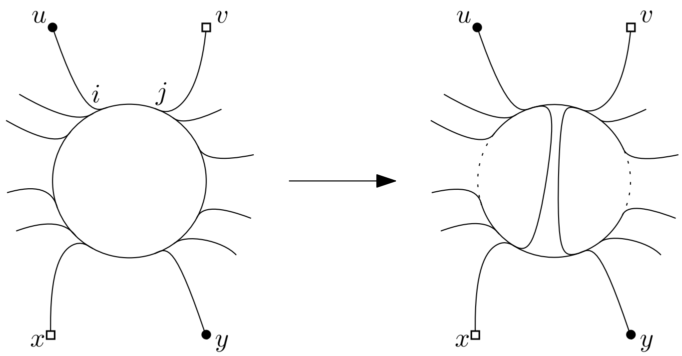


Figure 9: A circular path with only two merge-split pairs can be redrawn without change of the node order.

Proof: Let $p, q \in P(D)$ be two distinct uv-paths in a reduced confluent diagram $D = (N, J, \Gamma)$. We find two minimal distinct sub-paths p' of p and q' of q between two junctions i, j of p and q.

First we assume that i is a split-junction of p, q and j a merge-junction of p, q. We claim that each of p' and q' must contain at least two junctions that form a merge-split pair. The argument is symmetric, so we focus on p'. If p' passes through no junction, the arc of p' can be removed since q' achieves the same connectivity, but this contradicts that D is reduced. If p' passes through exactly one junction i_1, then the arc of p' that does not split at i_1 can be removed without changing any node adjacencies in G_D. So p' must contain at least two junctions i_1 and i_2. Assume there would be no merge-split pair. This means, coming from i path p' passes through a sequence of split-junctions followed by a sequence of merge-junctions. But in that case the arc connecting the last split-junction with the first merge-junction can be removed. So there must be a merge-split pair on p' and similarly on q'.

Next we follow each arc of these two merge-split pairs that does not lie on p' and q' towards some reachable node. This yields four nodes $x, y, w, z \in N$ that together with u and v form a domino subgraph as in Lemma 4, which is in fact a C_6 with a chord.

Assume there are no two uv-paths as in the first case. Then i is a merge-junction of p, q and j is a split-junction of p, q, see Figure 9. In this case, one of the paths, say q' contains a cycle and visits i and j twice. Clearly, all arcs of p' are essential as removing them would disconnect u and v. By the same arguments as before, there must be at least one merge-split pair on q' as otherwise we can delete an arc of q'. However, if there is a single minimal merge-split pair i_1, i_2 on q' (i.e., two directly adjacent junctions), then one can reroute the arcs joining q' in i_1 and i_2 towards i and j, respectively and remove two arcs from q', see Figure 8. Hence, there must be at least two merge-split pairs on q'. Consequently, we can find six nodes that form a $K_{3,3}$ subgraph as in Lemma 3, which is again a C_6 with at least one chord. □

Theorem 5 *The (bipartite-permutation ∩ domino-free)-graphs are exactly the strict bipartite-outerconfluent graphs.*

Proof: Every bipartite graph on four or five vertices is clearly domino-free as the domino graph contains six verticces. Moreover, they can all be drawn strict bipartite-outerconfluent as they either have an outerplanar drawing with the bipartite sets drawn as desired or they are isomorphic

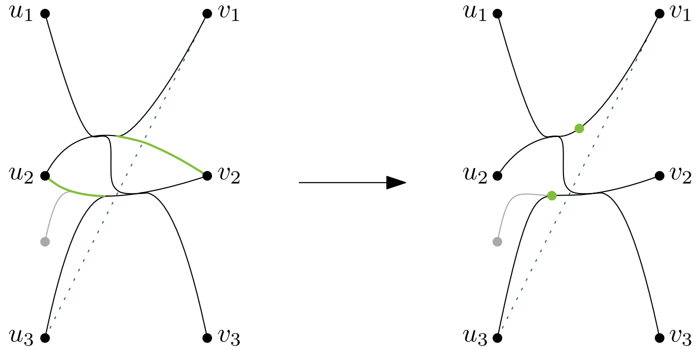


Figure 10: Redrawing a $K_{3,3}$ minus an edge. The blue dotted edge is missing. Since the graph is bipartite, we find that for every path using one of the thick green arcs on the left, we can redraw it such that it merges into the path coming from u_3 at the green marker in the middle without creating any wrong adjacencies.

to a cycle on four vertices, $K_{2,3}$ minus an edge, or $K_{2,3}$. All these graphs admit straight-forward strict bipartite-outerconfluent drawings as shown in Figure 11.

Let $G = (V, E)$ with $|V| \geq 6$ be a (bipartite-permutation $\cap$ domino-free) graph. By Theorem 4 we can find a bipartite-outerconfluent diagram $D = (N, J, \Gamma)$ which has $G_D = G$. Now assume that D is reduced but not strict. In this case we find six nodes $N' \subseteq N$ corresponding to a vertex set $V' \subseteq V_D$ in G_D such that $G_D[V'] = (V', E')$ is a C_6 with at least one chord by Lemma 5. In addition, since D (and hence also G_D) is bipartite and domino-free, we know there are two or three chords. But then $G_D[V']$ is just a $K_{3,3}$ minus one edge $e \in E'$ or $K_{3,3}$. In a bipartite diagram these can always be drawn in a strict way.

Let $V' = \{u_1, u_2, u_3, v_1, v_2, v_3\}$ where the u_i are on one side of the diagram and the v_i on the other; they are ordered by their indices from top to bottom. First, observe that since G is a bipartite permutation graph and the algorithm by Hui et al. [41] uses the strong ordering on the vertices, we get that in a $K_{3,3}$ minus an edge the missing edge is either u_1v_3 or u_3v_1. Further we can assume that the non-strict doubled path is between u_2 and v_2 since D is reduced. If this was not the case we would find two merge-split pairs with their vertices all below or above u_2, v_2 in D, but then one of these pairs has junctions with both uv-paths and we can reduce it.

Now, w.l.o.g., assume u_3v_1 is the missing edge. It follows that u_1v_3 exists as an edge and the u_1v_3 path must also have junctions with both u_2v_2 paths. Furthermore, we know both these junctions are merge junctions for u_1 and u_2, v_2 and v_3 respectively. Thus we can redraw as in Figure 10. For the case of no edge from $K_{3,3}$ missing the same argument applies.

For the other direction, consider a strict bipartite-outerconfluent diagram $D = (N, J, \Gamma)$. By Theorem 4, G_D is a bipartite permutation graph, and by Lemma 4, it must be domino free. Thus, G_D must be as described. □

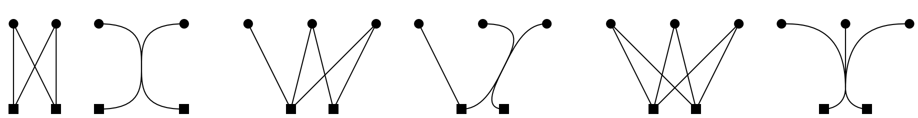


Figure 11: Strict bipartite-outerconfluent drawings of C_4, $K_{2,3}$ minus an edge, and $K_{2,3}$.

6 SOC Graphs Have Cop-Number Two

The *cops-and-robbers* game [1] on a graph $G = (V, E)$ is a two-player game with perfect information. The *cop-player* controls k *cop tokens*, while the *robber-player* has one *robber token*. In the first move the cop-player places the cop tokens on vertices of the graph, and then the robber places his token on another vertex. In the following the players alternate, in each turn moving their tokens to a neighboring vertex or keeping them at the current location. The cop-player is allowed to move all cops at once and multiple cops may be at the same vertex. The goal of the cop-player is to catch the robber, i.e., place one of its tokens on the same vertex as the robber. Note that, since it is a game with perfect information, the players can see each others token at all times.

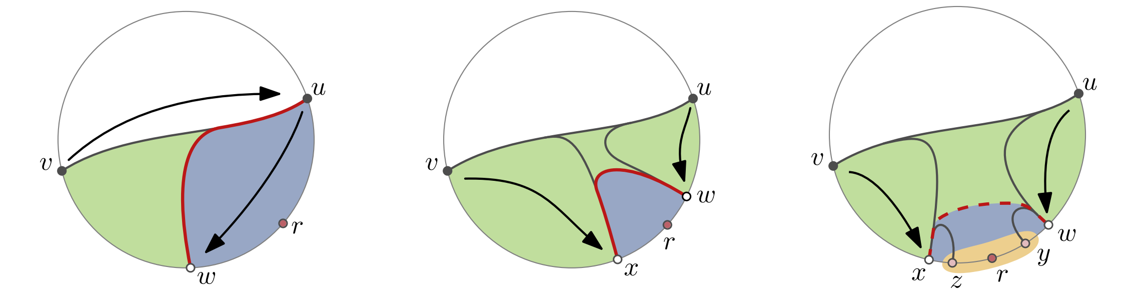
The *cop-number* $cop(G)$ of a connected graph G is the smallest integer k such that the cop-player has a winning strategy using k cop tokens. In case G is not connected $cop(G)$ is the maximum cop-number of any of its connected components [8]. Gavenčiak et al. [28] showed that the cop-number of outer-string graphs is between three and four, while the cop-number of many other interesting classes of intersection graphs, such as circle graphs and interval filament graphs, is two. We show that the cop-number of SOC graphs is two as well.

Consider an SOC drawing $D = (N, J, \Gamma)$ of a graph $G = (V, E)$. Note that if $G = (V, E)$ is not connected, each connected component can be treated as discussed below. Hence, the cop number is then just two times the number of connected components. For nodes $u, v \in N$, let the node interval $N[u, v] \subset N$ be the set of nodes in clockwise order between u and v on the outer face, excluding u and v. Let the cops be located on nodes $C \subseteq N$ and the robber be located on $r \in N$. We say that the robber is *locked* to a set of nodes $N' \subset N$ if $r \in N'$ and every path from r to $N \setminus N'$ contains at least one node that is either in C or adjacent to a node in C; in other words, a robber is locked to N' if it can be prevented from leaving N' by a cop player who simply remains stationary unless the robber can be caught in a single move. The following lemma establishes that a single cop can lock the robber to one of two “sides” of an SOC drawing.

Lemma 6 *Let $D = (N, J, \Gamma)$ be an SOC diagram of a graph G. Let a cop be placed on node u, the robber on node $r \neq u$ and not adjacent to u, and let $v \neq r$ be an arbitrary neighbor of u. Then the robber is either locked to $N[u, v]$ or locked to $N[v, u]$.*

Proof: Assume that, w.l.o.g., $r \in N[u, v]$, and consider an arbitrary path P in G from r to $N \setminus (N[u, v] \cup \{u, v\})$, which contains neither u nor a neighbor of u. Consider the first edge xy on P such that $y \notin N[u, v]$, and consider the xy-path in D. Since $y \neq u$ and $y \neq v$, it must hold that the xy-path in D crosses the uv-path at some junction. Hence, x must either be adjacent to u or to v; in the former case, this immediately contradicts our assumption that P contains no neighbor of u. In the latter case, it follows that there must be a junction on the uv-path in D, which is used by the xy-path to reach y, and hence u must also be adjacent to y—once again contradicting our initial assumption about P. □

Let $u, v \in N$ be two nodes of an SOC diagram $D = (N, J, \Gamma)$. We call a neighbor w of u in $N[u, v]$ *cw-extremal* (resp. *ccw-extremal*) for u, v (assuming such a neighbor exists), if it is the last neighbor of u in the clockwise (resp. counterclockwise) traversal of $N[u, v]$. Now let u, v be two adjacent nodes in N, $w \in N[u, v]$ be the cw-extremal neighbor for u and $x \in N[u, v]$ be the ccw-extremal neighbor for v. If w appears before x in the clockwise traversal of $N[u, v]$ we call w, x the *extremal pair* of the uv-path, see Figures 12b and 12c. In the case where only one node of u, v has an extremal neighbor w, say u, we define the extremal pair as v, w. In the following we assume that for a given uv-path the extremal pair exists.



(a) The cops confine the robber to $N[u, v]$.

(b) The cops confine the robber to $N[w, x]$.

(c) The cops confine the robber to $N[w, x]$.

Figure 12: Moves of the cops to confine the robber to a strictly smaller range.

Lemma 7 *Let $D = (N, J, \Gamma)$ be an SOC diagram of a graph G, $u, v \in N$ be two nodes connected by a uv-path in $P(D)$ and $w, x \in N[u, v]$ the extremal pair of the uv-path. If the cops are placed at u and v and the robber is at $r \in N[u, v]$, $r \neq w$, $r \neq x$, there is a move that locks the robber to $N[u, w]$, $N[w, x]$ or $N[x, v]$.*

Proof: In case $r \in N[u, w]$ or $r \in N[x, v]$ we can swap the cops in one move (see Figure 12a) by moving the cop from v to u and from u to w in the former case and from u to v and v to x in the latter. This locks the robber to $N[u, w]$ or $N[x, v]$ by Lemma 6.

The remaining case is $r \in N[w, x]$. By construction of the extremal pair, no $y \in N[w, x]$ is a neighbor of u or v. Because G is a connected graph, there must be at least one $y \in N[w, x]$ that is a neighbor of w or x as the only smooth paths leaving $N[w, x]$ must share a merge junction with u or v on the paths towards w or x. In the next step, we move the cops from u and v to w and x. We need to distinguish two subcases. If r is not a neighbor of w or x, then this position obviously locks the robber to $N[w, x]$ as any path leaving $N[w, x]$ must pass through a neighbor of w or x. If, however, r is already at a neighbor of $w \neq u$ or $x \neq v$, it may escape from $N[w, x]$ in the next move to a node in $N[u, w]$ or $N[x, v]$. But then by Lemma 6 it locks itself to $N[u, w]$ or $N[x, v]$. Note that if $w = u$ or $x = v$, then there is no way for r to escape across w or x, respectively, as r would be a neighbor of u or v in that case. □

Lemma 8 *Let $D = (N, J, \Gamma)$ be an SOC diagram of a graph G, $u, v \in N$ be two nodes connected by a uv-path in $P(D)$ and $w, x \in N[u, v]$ be the extremal pair of the uv-path such that there is no wx-path in $P(D)$. If the robber is at $r \in N[w, x]$ and the cops are placed on w, x we can find $y, z \in N[w, x] \cup \{w, x\}$ such that the yz-path exists in $P(D)$ and the robber is locked to $N[y, z]$.*

Proof: First, assume that there is a path in G connecting w and x which passes only through nodes in $N[w, x]$; see Figure 12c. Let $y \in N[w, x]$ be the ccw-extremal neighbor of x. If $r \in N[y, x]$ we are done and by Lemma 6 we can move the cop from w to y as the cop at x suffices to lock the robber to $N[y, x]$.

Now let $r \in N[w, y]$ instead and move the cop from x to y. As in the proof of the previous lemma, there are two subcases. If r is not a neighbor of y, the new position of the cops locks the robber to $N[w, y]$. Otherwise, the robber might escape to $N[y, x]$ but immediately locks itself to $N[y, x]$ by Lemma 6 and we are done. We repeat this process of going to the ccw-extremal neighbor until we eventually lock the robber to some $N[y, z]$ where the yz-path is in $P(D)$.

Now assume that there is no path from w to x in G that passes only through nodes in $N[w,x]$. But then the only possibility for the robber to leave $N[w,x]$ is by passing through a neighbor of just one of w or x, say x. We keep the cop at x, which suffices to lock the robber to $N[w,x]$. We can thus safely move the other cop first from w to x and from there following a path from one ccw-extremal neighbor to the next until reaching a node y such that the robber is now locked to $N[y,x]$ by the two cops. If there is an xy-path in $P(D)$ we are done. Otherwise we are now in the first case of the proof since by definition of y there is a path from x to y in G passing only through nodes in $N[y,x]$. □

Combining Lemmas 6, 7 and 8 yields the result.

Theorem 6 *SOC graphs have cop-number two.*

Proof: Let $D = (N, J, \Gamma)$ be a strict-outerconfluent diagram of a (connected) graph G. Choose any uv-path in $P(D)$ and place the cops on u and v as initial turn. The robber must be placed on a node r that is either in $N[u,v]$ or in $N[v,u]$; by symmetry, let us assume the former. By Lemma 6, the robber is now locked to $N[u,v] \neq \emptyset$.

In every move we will shrink the locked interval until eventually the robber is caught. Let $w \in N[u,v]$ be the cw-extremal neighbor of u and let $x \in N[u,v]$ be the ccw-extremal neighbor of v, which, for now, we assume to exist both. If the clockwise order of u,v,w,x on the outer face is $u < x < w < v$ then the robber must be either locked to $N[u,w]$ or to $N[x,v]$ by Lemma 6, so we move the cops to the respective nodes u,w or v,x and recurse with a smaller interval. If the clockwise order is $u < w < x < v$ then w,x is an extremal pair and by Lemma 7 we can lock the robber to a smaller interval in the next move. In case it is locked to $N[u,w]$ or to $N[x,v]$ we move the cops accordingly and recurse by Lemma 6. If it is locked to $N[w,x]$, then either there is a wx-path in $P(D)$ and again Lemma 6 applies (Figure 12b) or there is no wx-path and Lemma 8 applies after moving the cops to w and x.

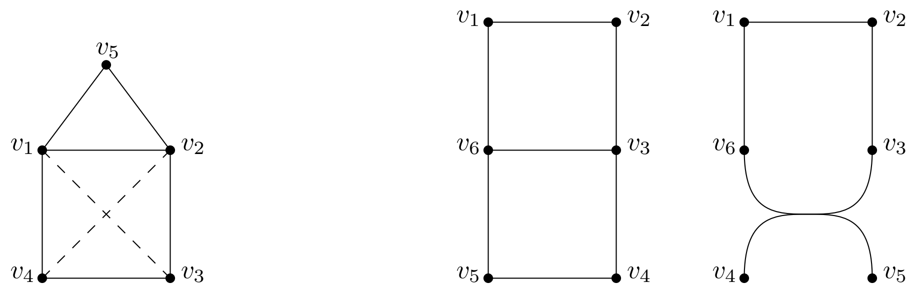
It remains to consider the case that u,v do not both have an extremal neighbor in $N[u,v]$. At least one of them, say u, must have a neighbor w in $N[u,v]$—otherwise G would not be connected. So we define the extremal pair as v,w. If the robber is in $N[u,w]$ we can move the cops to u,w as in Figure 12a and apply Lemma 6. If $r \in N[w,v]$, then we can move the cops to v,w and Lemma 8 applies. □

Theorem 6 suggests a closer link between SOC graphs and interval-filament graphs [30], another subclass of outer-string graphs with cop-number two.

7 Non-Inclusion Results for SOC Graphs

Here we collect smaller results for classes of graphs which have non-empty intersection with the class of SOC graphs, but are neither superclasses nor subclasses. Theorem 7 shows our incomparability results, while Corollary 3 lists classes which are not contained in the class of SOC graphs. Throughout this section, we denote by $C_n = (V,E)$ the cycle on n vertices. We assume that the vertices $V = \{v_1, \ldots, v_n, v_{n+1} = v_1\}$ of C_n are ordered such that $v_i v_{i+1} \in E$ for every $1 \leq i \leq n$. To simplify the notation we denote by $Z_\pi(G)$ the straight-line circular drawing of G with circular order π.

Recall that, given a graph $G = (V,E)$ and a circular ordering of the vertices, we say a crossing between edges $uv, wx \in E$ in the resulting straight line circular layout is representable if $G[\{u,v,w,x\}]$ has a $K_{2,2}$ subgraph.



(a) The house graph. (b) The domino graph.

Figure 13: The house and the domino graph.

Lemma 9 *Let $D = (N, J, \Gamma)$ be an outerconfluent diagram and π the circular ordering of the nodes in N. Then every induced cycle $C_n = \{v_1, \ldots, v_n\}$ with $n \geq 5$ in the corresponding outerconfluent graph $G_D = (V_D, E_D)$ appears in this or its reverse ordering in π.*

Proof: Let $D = (N, J, \Gamma)$ be an outerconfluent diagram of a graph $G_D = (V_D, E_D)$, $C_n = \{v_1, \ldots, v_n\}$ an induced cycle of G_D with more than five vertices, and π the cyclic ordering of the vertices V_D in D. Assume for contradiction that $v_1, \ldots, v_n$ do neither appear in order nor in reverse order of the indices. Then, we find a crossing between two edges of C_n in $Z_\pi(G_D)$. Though, since C_n has no chords, and consequently no induced C_4 or triangles, this crossing can never be representable as a confluent junction. Consequently, the outerconfluent diagram D does not exist by Lemma 2. □

Next we consider the *house graph*. It is a constant size graph on five vertices that consists of a C_4 and a triangle that has one edge identified with one edge in the cycle. Let $\mathcal{H} = (V_\mathcal{H}, E_\mathcal{H})$ be the house graph and $V = \{v_1, v_2, v_3, v_4, v_5\}$ its ordered set of vertices such that v_5 is the tip of the triangle and v_1, v_2, v_3, v_4 form the induced C_4, see Figure 13a.

Lemma 10 *Let G_D be an outerconfluent graph, $D = (N, J, \Gamma)$ an outerconfluent drawing of G_D, and π the inferred vertex order from D, the vertices $\{v_1, \ldots, v_5\}$ of every induced house graph $\mathcal{H}$ of G_D appear in order*

$$\langle v_1, v_5, v_2, v_3, v_4 \rangle \text{ or } \langle v_1, v_5, v_2, v_4, v_3 \rangle$$

in π.

Proof: It is easy to see that the two orderings have strict outerconfluent drawings. Hence, they can appear as sub-diagrams in an outerconfluent diagram of a graph containing an induced house graph. Let G_D be an outerconfluent graph containing induced houses and $D = (N, J, \Gamma)$ an outerconfluent drawing of G_D. Furthermore, let π be the vertex ordering of vertices in N inferred from D. Assume that $\mathcal{H} = (V_\mathcal{H}, E_\mathcal{H})$ is an induced house in G_D and its vertices are ordered not by one of the two above orders in π. This means we find a crossing between an edge v_5v_i for $i \in \{1, 2\}$ and an edge in $\{v_2v_3, v_3v_4, v_4v_1\}$ in $Z_\pi(G_D)$. This crossing is not representable, a contradiction to Lemma 2. □

We already considered the domino graph in Section 5. Here we show that the vertices of a domino graph have to be ordered according to one of six different orders to obtain a strict outerconfluent drawings. Note however, that these six orders encode only two non-isomorphic drawings. One is the outerplanar drawing and the other one with one junction. Both drawings are shown in Figure 13b. Let $\mathcal{D} = (V_{\mathcal{D}}, E_{\mathcal{D}})$ be the domino graph and let $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ be its ordered vertex set.

Lemma 11 *Let G_D be an outerconfluent graph, $D = (N, J, \Gamma)$ an outerconfluent drawing of G_D, and π the inferred vertex order from D, the vertices $\{v_1, \ldots, v_6\}$ of every induced domino graph $\mathcal{D}$ of G_D appear in order*

$$\langle v_1, v_2, v_3, v_4, v_5, v_6\rangle \text{ or } \langle v_2, v_1, v_3, v_4, v_5, v_6\rangle \text{ or } \langle v_2, v_1, v_6, v_4, v_5, v_3\rangle \text{ or}$$
$$\langle v_1, v_2, v_3, v_5, v_4, v_6\rangle \text{ or } \langle v_1, v_2, v_6, v_5, v_4, v_3\rangle \text{ or } \langle v_1, v_2, v_6, v_4, v_5, v_3\rangle$$

in π.

Proof: Clearly all these orders admit outerconfluent drawings of the domino graph $\mathcal{D}$ and can hence be sub-diagrams of a larger outerconfluent diagram. Now let $D = (N, J, \Gamma)$ be an outerconfluent diagram of a graph G_D containing an induced domino $\mathcal{D}$ and let π be the from D inferred vertex ordering. Assume that in π we find the vertices of $\mathcal{D}$ ordered non-isomorphic to one of the above orderings. This means that there is a crossing between two edges $ab, cd \in E_D \setminus \{v_3v_6\}, a, b, c, d \in V_D$ such that $a, b \in \{v_1, v_2, v_3, v_6\}$ and $c, d \in \{v_3, v_4, v_5, v_6\}$. Yet, there cannot be a C_4 such that the two of the edges are ab and cd. Consequently, this crossing can never be representable, a contradiction to Lemma 2. □

Lemma 11 holds for general outerconfluent graphs. In fact if we consider strict outerconfluent graphs we get the following corollary, whose proof follows directly from Lemmas 4 and 11.

Corollary 1 *Let G_D be a strict outerconfluent graph, $D = (N, J, \Gamma)$ an outerconfluent drawing of G_D, and π the inferred vertex order from D, the vertices $\{v_1, \ldots, v_6\}$ of every induced domino graph $\mathcal{D}$ of G_D appear in order*

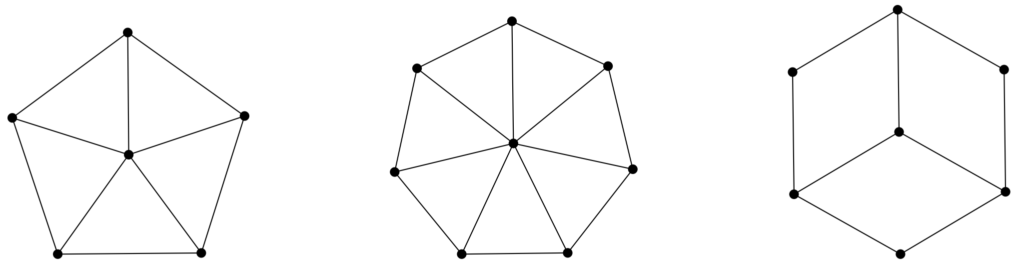
$$\langle v_1, v_2, v_3, v_4, v_5, v_6\rangle \text{ or } \langle v_2, v_1, v_3, v_4, v_5, v_6\rangle \text{ or } \langle v_2, v_1, v_6, v_4, v_5, v_3\rangle \text{ or}$$
$$\langle v_1, v_2, v_3, v_5, v_4, v_6\rangle \text{ or } \langle v_1, v_2, v_6, v_5, v_4, v_3\rangle$$

in π.

Before considering more complex classes of graphs we investigate two simple cases in the next lemma. It was already observed by Eppstein et al. [22] that the wheel on five vertices W_5 which is the graph obtained from a C_5 by adding one apex vertex, has no outerconfluent drawing. We show that this straight-forwardly holds for any $n \geq 5$. See Figures 14a and 14b for examples with $n = 5$ and $n = 7$. We note that the lemma also follows directly as outerconfluence is closed under taking induced subgraphs. The following proof is provided as Eppstein et al. do not provide a formal argument and we believe the observations made by our proof can be valuable in the future.

Lemma 12 *The wheel W_n has no outerconfluent drawing for $n \geq 5$.*

Proof: Assume $D = (N, J, \Gamma)$ is an outerconfluent drawing with $|N| - 1 = n \geq 5$ such that G_D is isomorphic to W_n. Clearly G_D contains an induced cycle C_{n-1} on $n-1$ vertices and by Lemma 9 this cycle has just one outerconfluent drawing. Now, at whatever position the apex vertex is going to be placed produces a crossing between all but two of its incident edges with the C_{n-1}. Let



(a) W_5 (b) W_5 (c) BW_3

Figure 14: The graphs W_5, W_7 and BW_3 used in Definition 1.

$v_a \in V_D$ be the apex vertex and let v_a be placed between v_{n-1} and v_1 along C_{n-1}. Furthermore, let $j = \lceil \frac{n-1}{2} \rceil$. Consider the crossing between the edge $v_{n-1}v_1$ and $v_a v_j$. This crossings can never be representable since it would require the existence of at least one chord in C_{n-1}. Our choice of v_1 and v_{n-1} was not a restriction, since we can shift the indices of the vertices in the cyclic order of C_{n-1},. Hence, by Lemma 2 there is no outerconfluent drawing of W_n with $n \geq 5$. □

The graph BW_3 is the graph obtained from a C_6 by adding a degree three vertex and connecting it to every second vertex in the C_6. Then the *bipartite wheel graph* [5] BW_n for $n \geq 4$ is obtained by combining a degree n vertex with a C_{2n} in the same way. In both classes we call the high-degree vertex the *central* vertex of the graph. Figure 14c shows BW_3.

Corollary 2 *The graph BW_n has no outerconfluent drawing for $n \geq 3$.*

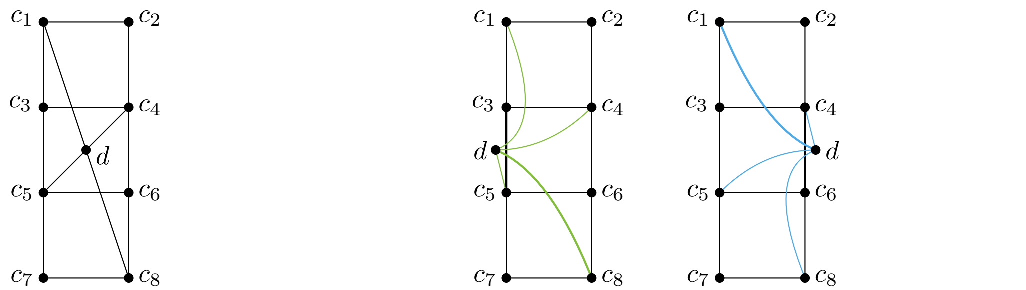
Proof: This follows directly from the proof of Lemma 12 □

These graphs are also relevant in the course of recognizing circle graphs. We now continue by giving short definitions for all the classes not yet defined in any previous section of this paper. As common in the graph theory literature we denote by *H-free graphs*, for a fixed graph H, the class of graphs that does not contain H as an induced subgraph.

Circle graphs are commonly defined as the class of intersection graphs that can be obtained from the intersections of chords in a circle. Here, we also use the following characterization of circle graphs due to Bouchet.

Definition 1 (Bouchet [9]) *The* local complement *$G * v$ is obtained from G by complementing the edges induced by v and its neighborhood in G. Two graphs are said to be* locally equivalent *if one can be obtained from the other by a series of local complements. A graph G is a circle graph if and only if no graph locally equivalent to G has an induced subgraph isomorphic to W_5, BW_3, or W_7 (see Figure 14).*

Circular-arc graphs are the graphs which have an intersection model of arcs of one circle [35]. Let ab, cd be two non-crossing chords of a circle and a, b, c, d points on the circle in order a, b, c, d. A *circle trapezoid* then consists of the two chords ab, cd and the circular-arcs bc and da. A *circle-trapezoid* graph is a graph which can be represented by intersecting circle trapezoids of one circle [24]. *Co-comparability* graphs are the intersection graphs of x-monotone curves in a vertical strip [33]. The *polygon-circle* graphs are the graphs which have an intersection model of polygons inscribed in the same circle [43]. *Interval filament* graphs, defined by Gavril [30], are intersection



(a) The graph.

(b) The final case in Lemma 13 the thick marked edges form a non-representable crossing.

Figure 15: Illustration of the counterexample for bipartite permutation ⊆ SOC.

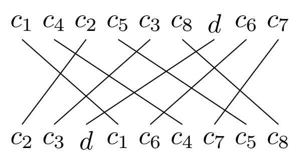


Figure 16: An intersection representation of the graph in Figure 15.

graphs of continuous, non-negative graphs of functions defined on closed intervals, such that they are zero-valued at their endpoints.

The *comparability* graphs are the transitively orientable graphs. We also use the forbidden subgraph characterization by Gallai [27]. The *alternation* graphs are the graphs which have a *semi-transitive orientation* [37]. For a graph $G = (V, E)$ a semi-transitive orientation is acyclic and for any directed path $v_1, \ldots, v_k$ we either find $v_1 v_k \notin E$ or $v_i v_j \in E$ for all $1 \leq i < j < k$.

The *chordal* graphs are the graphs, which have no chordless induced cycle of length at least four [32, 36]. *Co-chordal* graphs are the complement graphs of chordal graphs. *Series-parallel* graphs [17] are the graphs with terminal vertices s and t which can be constructed inductively from a single edge st, which is a series-parallel graph, using two composition operations. In either we are given two series-parallel graphs G_1 with terminals s_1 and t_1 and G_2 with terminals s_2 and t_2. In a *parallel* composition of G_1 and G_2 we take the disjoint union of G_1 and G_2 and and then identify s_1 with s_2 and t_1 with t_2. The two vertices resulting from this identification are the terminal vertices of the new series-parallel graph. In a *series* decomposition of G_1 and G_2 we identify s_2 with t_1 and take s_1 and t_2 as the terminal vertices of the new series-parallel graph.

Finally the class of *pseudo-split* graphs contains the graphs $G = (V, E)$ such that V can be partitioned into three sets C (a complete graph), I (an independent set) and S (if not empty an induced C_5). Further every vertex in C is adjacent to every vertex in S and every vertex in I is non-adjacent to every vertex in S.

We begin by showing that not every bipartite permutation graph can be drawn strict outerconfluent, complementing the result in Section 5. While the graph is small we did not find it by hand, but using an implementation of the algorithm for testing strict outerconfluency for a given vertex order by Eppstein et al. [22] which was recently created by Buchta in the course of his Master's thesis [10].

Lemma 13 *There is a bipartite permutation graph that is not an SOC graph.*

Proof: The graph depicted in Figure 15a is clearly bipartite. Moreover, the intersection diagram in Figure 16 encodes the same graph. Hence, it is a bipartite permutation graph.

It remains to show that the depicted graph is not a SOC graph. Let G be the graph and identify the vertices with their names in Figure 15a. Assume $D = (N, J, \Gamma)$ is a strict outerconfluent diagram such that G_D is isomorphic to G and let π be the ordering of the vertices. By Corollary 1 we know that the two induced domino subgraphs $C_1 = \{c_1, c_2, c_3, c_4, c_5, c_6\}$ and $C_2 = \{c_3, c_4, c_5, c_6, c_7, c_8\}$ each induce either an outerplane drawing or exactly two junctions in D (that is one representable crossing in $Z_\pi(G_D)$). Observe that the edges c_ic_{i+1} for $i = 1, 3, 5, 7$ are not crossed in $Z_\pi(D)$. Moreover, the vertices $c_1, \ldots, c_8$ appear in $Z_\pi(D)$ under the following restrictions in clockwise order starting from c_1 if it is followed by c_2 and else from c_2. $\{c_1, c_2\}$ appear first in some order, followed by c_3 or c_4, followed by c_5 or c_6, followed by $\{c_7, c_8\}$ in some order, followed by the other vertex of $\{c_5, c_6\}$, and finally the other vertex of $\{c_3, c_4\}$.

It remains to consider where d lies in $Z_\pi(D)$. First, observe that for each edge c_ic_{i+1} with $i = 1, \ldots, 8$ and $c_9 = c_1$ either dc_1 or dc_8 cannot cross that edge without creating a non-representable crossing in $Z_\pi(D)$. Consequently, d cannot be placed between c_1 and c_2 or c_7 and c_8 as this leads to a crossing between dc_8 and c_1c_2 or dc_1 and c_7c_8, respectively. Moreover, d cannot be placed directly after or before the two vertices in $\{c_1, c_2\}$ as this results in a crossing between dc_8 and c_3c_4. Analogously not directly before or after $\{c_7, c_8\}$ using dc_1 and c_5c_6. Finally, placing d in any of the two remaining positions leads to a crossing between dc_1 and c_4c_6 or dc_8 and c_3c_5 with either possibility resulting in a non-representable crossing. The latter cases are illustrated in Figure 15b. □

The main theorem of this section shows a series of non-inclusion results for most of the previously defined classes.

Theorem 7 *These graph classes are incomparable to SOC graphs: bipartite permutation, circle, circular-arc, (co-)chordal, (co-)comparability, domino-free, pseudo-split and series-parallel.*

Proof: We begin by showing the statement for bipartite permutation graphs.

Bipartite Permutation. By Lemma 13 there exists a graph that is a bipartite permutation graph, but not an SOC graph. C_5 is outerplanar and hence an SOC graph, but is not a bipartite permutation graph.

Bipartite permutation graphs are permutation graphs and hence also circle, comparability, co-comparability, and trapezoid graphs by definition. Hence for all these classes it also holds that not every graph is an SOC graph. It remains to show the converse for each.

Circle. Using the characterization of circle graphs due to Bouchet [9], we show that the graph in Figure 17 is not a circle graph.

Comparability. The graph in Figure 18 has a strict outerconfluent drawing, but is among the forbidden subgraphs of the class of comparability graphs [27].

Co-comparability. C_5 is not a co-comparability graph, but it is an SOC graph.

Trapezoid. C_5 is not a trapezoid graph, but is an SOC graph.

Next, we show the statement for chordal and co-chordal graphs, as well as the domino-free graphs. We use an argument that was already noticed by Dickerson et al. [16]. Let $K_n = (V, E)$ be a complete graph with $n \geq 5$ and attach for every $uv \in E$ a vertex w with edges uw and wv. This graph is chordal, but it does not even have a confluent drawing as the added vertices are not part of any C_4. We observe that in the case of outerconfluent drawings this already holds for K_4.

Chordal. C_4 is not a chordal graph by definition, but it is an SOC graph. As argued above there is a graph that is chordal but not outerconfluent.

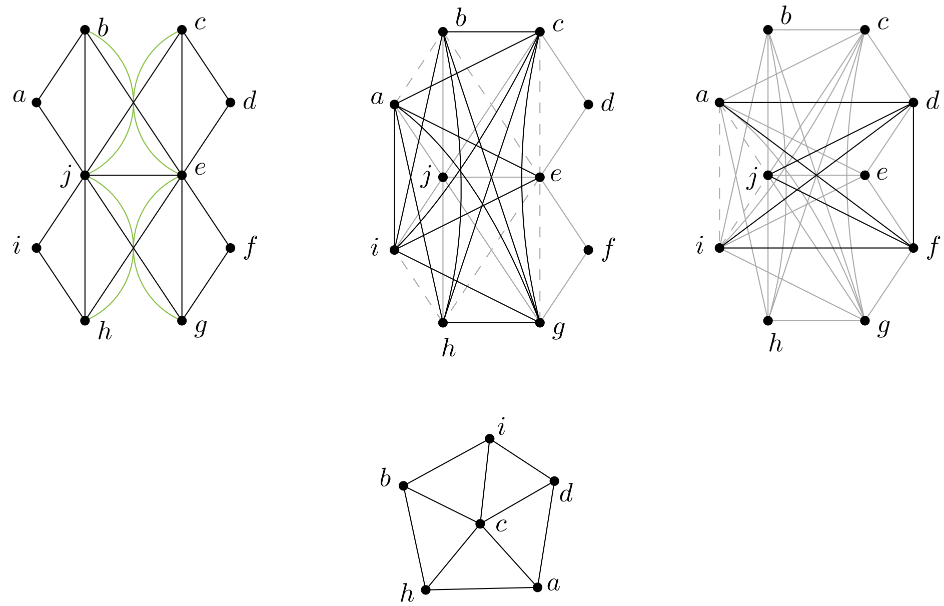


Figure 17: Counterexample for SOC $\subseteq$ circle. The green edges give the SOC drawing.

Co-chordal. C_5 is self-complementary and has a strict outerconfluent drawing, but co-chordal graphs do not contain the complement of C_{n+4}. Take the complement of a K_4 to which we added vertices as above, let $\overline{K}$ be this graph. Notice that in $\overline{K}$ we find that the four vertices of the K_4 are independent and that there is a clique between the ones that we added. Observe that any vertex that was in the K_4 has degree three in $\overline{K}$. As there are four old and six new vertices we find at least one crossing between edges both incident to vertices in the K_4. This crossing can never be representable as by construction two K_4 vertices share exactly one common neighbor and have no edge between them. Hence, they are not part of any C_4.

Domino-free. Domino-free graphs contain chordal and co-chordal graphs as a domino contains a C_4 as well as a $\overline{C_4}$. Since there are chordal and co-chordal graphs which are not SOC graphs there is also a domino-free graph that is not an SOC graph. Conversely, the domino is of course an outerplanar graph and consequently an SOC graph.

The remaining classes are circular-arc, pseudo-split and series-parallel graphs. The first two results make use of the fact that W_5 is no strict outerconfluent graph by Lemma 12.

Circular-arc. Circular-arc graphs do not contain every complete bipartite graph, but obviously those have a strict outerconfluent drawing. Conversely this class contains W_5.

Pseudo-split. C_4 is not a pseudo split graph, but it is an SOC graph. W_5 is a pseudo-split since we can take the central vertex as the clique and the other five vertices as the C_5.

Series-parallel. K_4 is not a series-parallel graph by definition, but it is an SOC graph. Let $G = (V, E)$ be a domino graph. Let the edge $uv \in E$ be the chord of the domino graph. (For comparison, this is the edge between v_6 and v_3 in Figure 13b.) Subdivide uv with a vertex w. This is a series-parallel graph, but any crossing between uw or vw and another edge cannot be represented. □

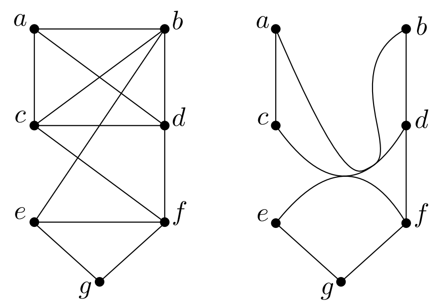


Figure 18: Counterexample for SOC $\subseteq$ comparability.

Corollary 3 *The following graph classes are not contained in the class of SOC graphs: alternation, BW_3-free, circle-trapezoid, polygon-circle, interval-filament, subtree-filament, (outer-)string.*

Proof: Comparability graphs are contained in alternation graphs [37], but comparability graphs are not a subclass of SOC graphs. All other classes follow directly, since they are known to be superclasses of circle graphs. □

The classes in Theorem 7 seem arbitrary to some degree. As a motivation we list a few intuitively interesting consequences. The relation to the domino-free graphs is of interest, as the existence of two non-strict paths in a strict outerconfluent diagram implies a twisted domino by Lemma 4. While it is clear that not all SOC graphs are domino-free it is interesting to see if all domino-free graphs are drawable strict outerconfluent. For the bipartite permutation graphs we confirmed this as they, by Theorem 5, have a strict-outerconfluent drawing when dominos are forbidden. This makes, to our knowledge, the graph in Figure 15 the first graph to be shown to have no strict outerconfluent drawing, but an outerconfluent one. The relation to the cop-number shown in Theorem 6 hints at a relation of SOC graphs with geometric intersection graphs of the same cop-number [28]. The above study shows that most classes "surrounding" interval-filament graphs are all incomparable to SOC graphs, which raises the natural question if all SOC graphs are interval filament graphs. In fact there is a curious inclusion path left over that we were not able to show incomparability for. These classes lie between the incomparable class of circle graphs and the superclass of outer-string graphs.

$$\text{circle} \subset \text{circle-trapezoid} \subset \text{polygon-circle} \subseteq \text{interval-filament} \subset \text{outer-string}.$$

To sum up, our study shows that many geometric intersection graphs do not admit strict outerconfluent drawings. At the same time we could identify classes that form non-trivial superclasses of SOC graphs.

8 Clique-width of Tree-like SOC Graphs

In 2005, Eppstein et al. [21] showed that every strict outerconfluent graph whose arcs in a strict outerconfluent drawing topologically form a tree is distance hereditary and hence exhibits certain well-understood structural properties—in particular, every such graph has bounded *clique-width* [12]. More precisely, distance hereditary graphs are equivalent to Δ-*confluent* graphs, which have a tree-like confluent drawing using an additional 3-way junction, called a Δ-*junction*, that smoothly links together all three incident arcs. See Figure 19, where the junctions j' and k' now form a single Δ-junction instead of three separate merge or split junctions.

In this section, we lift the result of Eppstein et al. [21] to the class of strict outerconfluent graphs whose arcs may not form a topological tree: in particular, we show that as long as the arcs incident to junctions (including Δ-junctions) topologically form a tree, strict outerconfluent graphs also have bounded clique-width. Equivalently, we show that "extending" any drawing covered by Eppstein et al. [21] through the addition of outerplanar drawings of graphs on a subset of the vertices in order to produce a strict outerconfluent drawing does not substantially increase the clique-width of the graph. Since the notion of clique-width will be central to this section, we formally introduce it below (see also the work of Courcelle et al. [12]).

A k-graph is a graph whose vertices are labeled by $[k] = \{1, 2, \dots, k\}$; formally, the graph is equipped with a labeling function $\gamma \colon V(G) \to [k]$, and we also use $\gamma^{-1}(i)$ to denote the set of vertices labeled i for $i \in [k]$. We consider an arbitrary graph as a k-graph with all vertices labeled by 1. We call the k-graph consisting of exactly one vertex v labeled by $j \in [k]$ an *initial* k-graph and denote it by $j(v)$. The clique-width of a graph G is the smallest integer k such that G can be constructed from *initial* k-graphs by means of repeated application of the following three operations:

1. Disjoint union (denoted by $\oplus$);
2. Relabeling: changing all labels i to j (denoted by $p_{i \to j}$);
3. Edge insertion: adding an edge between every vertex labeled by i and every vertex labeled by j, where $i \neq j$ (denoted by $\eta_{i,j}$ or $\eta_{j,i}$).

The construction sequence of a k-graph G using the above operations can be represented by an algebraic term composed of $i(v)$, $\oplus$, $p_{i \to j}$ and $\eta_{i,j}$ (where $v \in V(G)$, $i \neq j$ and $i, j \in [k]$). Such a term is called a *k-expression* defining G, and the *clique-width* of G is the smallest integer k such that G can be defined by a k-expression. Distance-hereditary graphs are known to have clique-width at most 3 [34] and outerplanar graphs have clique-width at most 5 due to having treewidth at most 2 [4, 13]. As an example illustrating the notation, the 2-expression $\eta_{1,2}((1(a) \oplus 1(b)) \oplus (2(c) \oplus 2(d)))$ demonstrates that the graph $K_{2,2}$ has clique-width at most 2.

Let *(tree-like)* Δ*-SOC graphs* be the class of all graphs which admit strict outerconfluent drawings (including Δ-junctions) such that the union of all arcs incident to at least one junction topologically forms a tree. Clearly, the edge set E of every tree-like Δ-SOC graph $G = (V, E)$ with confluent diagram D_G can be partitioned into sets E_s and E_c, where E_s (the set of *simple edges*) contains all edges represented by single-arc paths in D not passing through any junction and E_c (the set of confluent edges) contains all remaining edges in G. Let $G_c = G[E_c] = (V_c, E_c)$ be the subgraph of G induced by E_c, i.e., V_c is obtained from V by removing all vertices without incident edges in E_c.

We note that even though G_c is known to be distance-hereditary [21] and $G - E_c$ is easily seen to be outerplanar, this does not imply that tree-like Δ-SOC graphs have bounded clique-width—indeed, the union of two graphs of bounded clique-width may have arbitrarily high clique-width (consider, e.g., the union of two sets of disjoint paths that create a square grid). Furthermore, one cannot easily adapt the proof of Eppstein et al. [21] to tree-like Δ-SOC graphs, as that explicitly uses the structure of distance-hereditary graphs; note that there exist outerplanar graphs which are not distance-hereditary, and hence tree-like Δ-SOC graphs are a strict superclass of distance hereditary graphs. Before proving the desired theorem, we introduce an observation which will later allow us to construct parts of G in a modular manner.

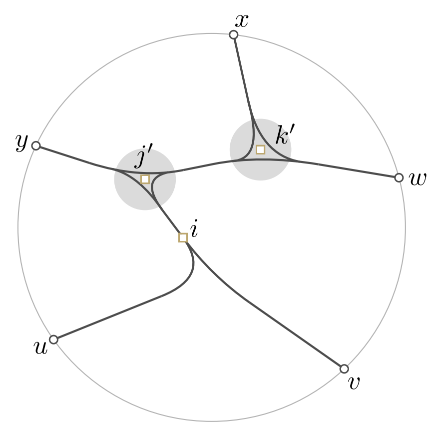


Figure 19: A Δ-confluent diagram representing $K_5 - (u, v)$. Nodes are disks, junctions are squares. Δ-junctions are marked with a grey circle.

Lemma 14 *Let $H = (V, E)$ be a graph of clique-width $k \geq 2$, let V_1, V_2 be two disjoint subsets of V, and let $s \in V \setminus (V_1 \cup V_2)$. Then there exists a $(3k+1)$-expression defining H so that in the final labeling all vertices in V_1 receive label 1, all vertices in V_2 receive label 2, s receives label 3 and all remaining vertices receive label 4.*

Proof: Consider an arbitrary k-expression of H which ends by setting all labels in H to 1. Now adjust the k-expression as follows: whenever a vertex in V_1 receives a label i, replace it with $i + k$, and whenever a vertex in V_2 receives a label i, replace it with $i + 2k$, and use a special label $3k + 1$ for s. This new $(3k+1)$-expression constructs H and assigns all vertices in $V \setminus (V_1 \cup V_2 \cup \{s\})$, V_1, V_2 and $\{s\}$ the labels 1, $k+1$, $2k+1$, and $3k+1$, respectively. To complete our construction, we merely map label 1 to 4, label $k+1$ to 1, label $2k+1$ to 2, and label $3k+1$ to 3. □

Theorem 8 *Every tree-like Δ-SOC graph has clique-width at most* 16.

Proof: Let us consider an arbitrary tree-like Δ-SOC graph $G = (V, E)$ and let us fix a tree-like strict outerconfluent drawing $D = (N, J, \Gamma)$ of G; let Γ_c be the set of arcs with at least one endpoint in J. Based on D, we partition E into the edge sets E_c and E_s as above. Let V_c be the set of vertices incident to at least one edge in E_c and let $G_c = (V_c, E_c)$.

Note that $D_c = (N, J, \Gamma_c)$ is topologically equivalent to a tree (plus some singletons in $V \setminus V_c$). Our aim will be to pass through D_c in a leaves-to-root manner (whereas the root will be selected later) so that at each step we construct a 16-expression for a subgraph induced by a certain set of consecutive vertices on the outer circle. This way, we will gradually build up the 16-expression for G from modular parts, and once we reach the root we will have a complete 16-expression for G.

Our proof will perform induction along a notion of *height*, which is tied to the tree-like structure of D_c. We say that each node corresponding to a vertex in V_c has *height* 0, and we define the height of each junction j as follows: j has height ℓ if ℓ is the minimum integer such that at least two of the arcs incident to j lead to junctions or nodes of height at most $\ell - 1$ (for example, a junction of height 1 has 2 arcs leading to nodes, while a junction of height 2 has at least one arc leading to a junction of height 1 and the other arc leads to either a node or another junction of height 1). We call an arc between a junction j of height i and a junction (or node) of height smaller than i a *down-arc*

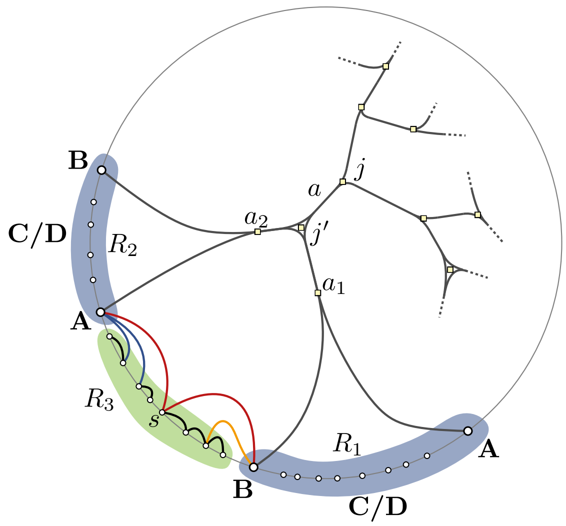


Figure 20: Sketch of a tree-like Δ-SOC graph G with the regions and junctions used in the inductive construction of the 16-expression defining G.

for j. To formally fix a root, we choose an arbitrary arc ab with a maximum combined height of its endpoints (i.e., such that no arc $a'b'$ satisfies $\text{height}(a)+\text{height}(b) < \text{height}(a')+\text{height}(b')$), and denote it as the *root arc* r.

Another notion we will use is that of a *region*, see Figure 20: the region defined by a junction j and one of its down-arcs a is the segment of the boundary of the outer face delimited by the "right-most" and "left-most" paths (not necessarily smooth), which leave j through a. Crucially, we observe that the set V_R of all vertices corresponding to nodes in a region R can be partitioned into the following four groups:

A. one vertex on the left border of R;

B. up to one vertex that is not on the left border but on the right border of R;

C. vertices not on the border which have no neighbors outside of R;

D. vertices not on the border which have at least one neighbor outside of R;

and furthermore we observe that all vertices of group **D** have precisely the same neighborhood outside of R (in particular, they must all have a path to j which forms a smooth curve). In the degenerate case of nodes (which have height 0), we say that the region is merely the point of that node (and the corresponding vertex then belongs to group **A**).

As the first step of our procedure, for each $v \in V_c$ we create a 1-expression $1(v)$ (i.e., we create each vertex in V_c as a singleton). For the second step, we apply induction along the height of junctions as follows. As our inductive hypothesis at step i, we assume that for each junction j' of height at most $i-1$ and each of its down-arcs defining a region R', there exists a 16-expression which constructs $G[V_{R'}]$ and labels $V_{R'}$ by using labels $1, 2, 3, 4$ for vertices in groups **A**, **B**, **C**, **D**, respectively. We observe that the inductive hypothesis holds at step 1: indeed, all regions at height 0 consist of a node, and we already created the respective 1-expressions for all such nodes.

Our aim is now to use the inductive hypothesis for i to show that the inductive hypothesis also holds for $i+1$—in other words, we need to obtain a 16-expression which constructs and correctly labels the graph $G[V_R]$ for the region R defined by each junction j of height i and down-arc a. Assume that a is incident to a junction j' with down-arcs a_1 and a_2, defining the regions R_1 and R_2, respectively. By our inductive assumption, $G[V_{R_1}]$ and $G[V_{R_2}]$ both admit a 16-expression which labels the vertices based on their group in the desired way. Now observe that R is composed of the following parts: region R_1 on the "left", region R_2 on the "right", and a segment R_3 on the boundary of the outer face between R_1 and R_2. Crucially, we make the following observations for vertices V_{R_3} in R_3:

- none of the vertices in V_{R_3} are incident to an edge in E_c;
- $G[V_{R_3}]$ is outerplanar;
- at most one vertex, denoted s, in V_{R_3} has two neighbors outside of V_{R_3}, notably the rightmost vertex in V_{R_1} and the leftmost vertex in V_{R_2};
- all vertices other than s in V_{R_3} either have no neighbors outside of V_{R_3}, or have one neighbor outside of V_{R_3}—in particular, either the rightmost vertex in V_{R_1} or the leftmost vertex in V_{R_2}.

At this point, we can finally invoke Lemma 14. In particular, since $G[V_{R_3}]$ is outerplanar, it has clique-width at most 5, and by using the observation we can construct a 16-expression which labels all vertices adjacent to the right border of R_1 with label 1, all vertices adjacent to the left border of R_2 with label 2, vertex s with label 3, and all other vertices with label 4. Now all that remains is to:

1. relabel labels 1–4 used in the 16-expression for R_2 to labels 5–8 and the labels 1–4 used in the 16-expression for R_3 to labels 9–12, respectively;
2. use the $\oplus$ operator to merge these 16-expressions,
3. use the $\eta_{i,j}$ operator to add edges between $V_{R_1} \cup V_{R_2}$ and V_{R_3} as required, in particular: $\eta_{9,2}$, $\eta_{11,2}$, $\eta_{10,5}$, $\eta_{11,5}$;
4. use the $\eta_{4,8}$ operator to add all pairwise edges between the groups **D** of V_{R_1} and V_{R_2} in case junction j' smoothly connects arcs a_1 and a_2;
5. use the $p_{i \to j}$ operator to relabel as required by the inductive assumption, where depending on the junction type of j' group **D** of V_R either consist of the union of the groups **D** of V_{R_1} and V_{R_2} or it is identical to group **D** of just one of them. Group **A** coincides with group **A** of V_{R_1} and group **B** coincides with group **B** of V_{R_2}. The remaining vertices form group **C**.

The inductive procedure described above runs until it reaches the root arc r, and it is easy to observe that at this point we have constructed two 16-expressions corresponding to the two regions, say R_1^* and R_2^*, defined by paths which start at r and go in the two possible directions. The two remaining regions on the outer face between R_1^* and R_2^* are then handled completely analogously as the regions denoted R_3 in our inductive step. Hence we conclude that there indeed exists a 16-expression which constructs G. □

9 Conclusion

In this paper we presented an in-depth study of strict outerconfluent graph drawings. While it was not possible to resolve the main open question whether recognizing graphs admitting such drawings is tractable or not, we could make, we hope, significant steps towards resolving it. More precisely, we showed that (outer-)string graphs contain the strict (outer-)confluent graphs. Furthermore, we showed that every unit interval graph is strict confluent and every domino-free bipartite permutation graph is strict outerconfluent. We also gave a wide variety of smaller results, excluding many geometric intersection graph classes from the list of possible sub- or super-classes of strict outerconfluent graphs. Complementing the results on string graphs we showed that strict outerconfluent graphs have cop-number two. As outer string graphs have cop-number at least three this hints that it should be possible to refine our superclass. Finally, we showed that tree-like Δ-confluent graphs, a generalization of the Δ-confluent graphs presented by Eppstein et al. [21], have bounded cliquewidth.

Among the intersection graph classes with cop-number two identified by Gavenčiak et al. [28] there are so-called interval filament graphs. We suspect that the strict outerconfluent graphs are in fact a sub-class of the interval filament graphs, but we have not been able to prove this conclusively. Similarly, it is open whether SC graphs are contained in subtree-filament graphs. Furthermore, it is conceivable that a similar construction for the inclusion in string graphs, Section 3, could be used to show similar results for non-strict confluent graphs. Finally, investigating the curve complexity of our construction might provide insight into the curve complexity of SC and SOC diagrams.

On the algorithmic side, Section 8 raises the question of whether clique-width might be used to recognize SOC graphs, and perhaps even for finding SOC drawings. Another decomposition-based approach would be to use so-called split-decompositions [31], which we did not consider here. It is also open whether it is possible to drop the unit length condition on the intervals in Section 4. We did not see an obvious way of adapting the construction for confluent drawings of interval graphs [16]. In the same direction, we could present a bipartite permutation graph that is not strict outerconfluent, yet it would be interesting to investigate if all bipartite permutation graphs are in fact strict confluent.

Acknowledgments

The authors thank Markus Buchta for the code used in initially obtaining the counterexample in Lemma 13.

References

[1] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984. doi:10.1016/0166-218X(84)90073-8.

[2] M. J. Alam, M. Fink, and S. Pupyrev. The Bundled Crossing Number. In Y. Hu and M. Nöllenburg, editors, *24th International Symposium on Graph Drawing and Network Visualization (GD'16)*, volume 9801 of *LNCS*, pages 399–412. Springer, 2016. doi:10.1007/978-3-319-50106-2_31.

[3] B. Bach, N. H. Riche, C. Hurter, K. Marriott, and T. Dwyer. Towards Unambiguous Edge Bundling: Investigating Confluent Drawings for Network Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):541–550, 2017. doi:10.1109/TVCG.2016.2598958.

[4] B. S. Baker. Approximation Algorithms for NP-Complete Problems on Planar Graphs. *Journal of the ACM*, 41(1):153–180, 1994. doi:10.1145/174644.174650.

[5] H. Bandelt. Hereditary modular graphs. *Combinatorica*, 8(2):149–157, 1988. doi:10.1007/BF02122796.

[6] H.-J. Bandelt and H. M. Mulder. Distance-hereditary graphs. *Journal of Combinatorial Theory, Series B*, 41(2):182–208, 1986. doi:10.1016/0095-8956(86)90043-2.

[7] C. Benzaken, Y. Crama, P. Duchet, P. L. Hammer, and F. Maffray. More characterizations of triangulated graphs. *Journal of Graph Theory*, 14(4):413–422, 1990. doi:10.1002/jgt.3190140404.

[8] A. Berarducci and B. Intrigila. On the cop number of a graph. *Advances in Applied Mathematics*, 14(4):389–403, 1993. doi:10.1006/aama.1993.1019.

[9] A. Bouchet. Circle graph obstructions. *Journal of Combinatorial Theory, Series B*, 60(1):107–144, 1994. doi:10.1006/jctb.1994.1008.

[10] M. Buchta. Engineering an algorithm for strict outerconfluent graph drawings. Master's thesis, TU Wien, 2019. URL: https://resolver.obvsg.at/urn:nbn:at:at-ubtuw:1-130995.

[11] S. Chaplick, T. C. van Dijk, M. Kryven, J.-w. Park, A. Ravsky, and A. Wolff. Bundled crossings revisited. In D. Archambault and C. D. Tóth, editors, *27th International Symposium on Graph Drawing and Network Visualization (GD'19)*, volume 11904 of *LNCS*, pages 63–77. Springer, 2019. doi:10.1007/978-3-030-35802-0_5.

[12] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. doi:10.1007/s002249910009.

[13] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1-3):77–114, 2000. doi:10.1016/S0166-218X(99)00184-5.

[14] H. Czemerinski, G. A. Durán, and A. Gravano. Bouchet graphs: A generalization of circle graphs. *Congressus Numerantium*, pages 95–108, 2002.

[15] I. Dagan, M. C. Golumbic, and R. Y. Pinter. Trapezoid graphs and their coloring. *Discrete Applied Mathematics*, 21(1):35–46, 1988. doi:10.1016/0166-218X(88)90032-7.

[16] M. Dickerson, D. Eppstein, M. T. Goodrich, and J. Y. Meng. Confluent Drawings: Visualizing Non-planar Diagrams in a Planar Way. *Journal of Graph Algorithms and Applications*, 9(1):31–52, 2005. doi:10.7155/jgaa.00099.

[17] R. J. Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10(2):303 – 318, 1965. doi:10.1016/0022-247X(65)90125-3.

[18] T. Dwyer, C. Mears, K. Morgan, T. Niven, K. Marriott, and M. Wallace. Improved optimal and approximate power graph compression for clearer visualisation of dense graphs. In I. Fujishiro, U. Brandes, H. Hagen, and S. Takahashi, editors, *IEEE Pacific Visualization Symposium (PacificVis'14)*, pages 105–112. IEEE, 2014. doi:10.1109/PacificVis.2014.46.

[19] T. Dwyer, N. H. Riche, K. Marriott, and C. Mears. Edge compression techniques for visualization of dense directed graphs. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2596–2605, 2013. doi:10.1109/TVCG.2013.151.

[20] D. Eppstein, M. T. Goodrich, and J. Y. Meng. Confluent Layered Drawings. *Algorithmica*, 47(4):439–452, 2007. doi:10.1007/s00453-006-0159-8.

[21] D. Eppstein, M. T. Goodrich, J. Y. Meng, P. Healy, and N. S. Nikolov. Delta-confluent drawings. In *13th International Symposium on Graph Drawing (GD'05)*, volume 3843 of *LNCS*, pages 165–176. Springer, 2005. doi:10.1007/11618058_16.

[22] D. Eppstein, D. Holten, M. Löffler, M. Nöllenburg, B. Speckmann, and K. Verbeek. Strict confluent drawing. *Journal of Computational Geometry*, 7(1):22–46, 2016. doi:10.20382/jocg.v7i1a2.

[23] D. Eppstein and J. A. Simons. Confluent Hasse Diagrams. *Journal of Graph Algorithms and Applications*, 17(7):689–710, 2013. doi:10.7155/jgaa.00312.

[24] S. Felsner, R. Müller, and L. Wernisch. Trapezoid graphs and generalizations, geometry and algorithms. *Discrete Applied Mathematics*, 74(1):13–32, 1997. doi:10.1016/S0166-218X(96)00013-3.

[25] M. Fink, J. Hershberger, S. Suri, and K. Verbeek. Bundled Crossings in Embedded Graphs. In E. Kranakis, G. Navarro, and E. Chávez, editors, *12th Latin American Symposium on Theoretical Informatics (LATIN'16)*, volume 9644 of *LNCS*, pages 454–468. Springer, 2016. doi:10.1007/978-3-662-49529-2_34.

[26] C. P. Gabor, K. J. Supowit, and W.-L. Hsu. Recognizing circle graphs in polynomial time. *Journal of the ACM*, 36(3):435–473, 1989. doi:10.1145/65950.65951.

[27] T. Gallai. Transitiv orientierbare Graphen. *Acta Mathematica Hungarica*, 18(1-2):25–66, 1967. doi:10.1007/bf02020961.

[28] T. Gavenčiak, P. Gordinowicz, V. Jelínek, P. Klavík, and J. Kratochvíl. Cops and Robbers on intersection graphs. *European Journal of Combinatorics*, 72:45–69, 2018. doi:10.1016/j.ejc.2018.04.009.

[29] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972. doi:10.1137/0201013.

[30] F. Gavril. Maximum weight independent sets and cliques in intersection graphs of filaments. *Information Processing Letters*, 73(5-6):181–188, 2000. doi:10.1016/S0020-0190(00)00025-9.

[31] E. Gioan and C. Paul. Split decomposition and graph-labelled trees: Characterizations and fully dynamic algorithms for totally decomposable graphs. *Discrete Applied Mathematics*, 160(6):708–733, 2012. doi:10.1016/j.dam.2011.05.007.

[32] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Elsevier, 2004.

[33] M. C. Golumbic, D. Rotem, and J. Urrutia. Comparability graphs and intersection graphs. *Discrete Mathematics*, 43(1):37 – 46, 1983. doi:10.1016/0012-365X(83)90019-5.

[34] M. C. Golumbic and U. Rotics. On the Clique-Width of Some Perfect Graph Classes. *International Journal of Foundations of Computer Science*, 11(3):423–443, 2000. doi:10.1142/S0129054100000260.

[35] H. Hadwiger and H. Debrunner. *Combinatorial Geometry in the Plane*. Dover Books on Mathematics, 2015.

[36] A. Hajnal and J. Surányi. über die Auflösung von Graphen in vollständige Teilgraphen. *Annales Universitatis Scientiarium Budapestinensis de Rolando Eötvös Nominatae Sectio Mathematica*, 1:113–121, 1958.

[37] M. M. Halldórsson, S. Kitaev, and A. Pyatkin. Alternation graphs. In P. Kolman and J. Kratochvíl, editors, *37th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'11)*, volume 6986 of *LNCS*, pages 191–202. Springer, 2011. doi:10.1007/978-3-642-25870-1_18.

[38] D. Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006. doi:10.1109/TVCG.2006.147.

[39] D. Holten and J. J. van Wijk. Force-directed edge bundling for graph visualization. *Computer Graphics Forum*, 28:983–990, 2009. doi:10.1111/j.1467-8659.2009.01450.x.

[40] W.-L. Hsu. Maximum weight clique algorithms for circular-arc graphs and circle graphs. *SIAM Journal on Computing*, 14(1):224–231, 1985. doi:10.1137/0214018.

[41] P. Hui, M. J. Pelsmajer, M. Schaefer, and D. Stefankovic. Train tracks and confluent drawings. *Algorithmica*, 47(4):465–479, 2007. doi:10.1007/s00453-006-0165-x.

[42] S. G. Kobourov. Force-directed drawing algorithms. In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization*, pages 383–408. Chapman and Hall/CRC, 2013.

[43] A. Kostochka and J. Kratochvíl. Covering and coloring polygon-circle graphs. *Discrete Mathematics*, 163(1-3):299–305, 1997. doi:10.1016/S0012-365X(96)00344-5.

[44] J. Kratochvíl. String Graphs. I. The Number of Critical Nonstring Graphs Is Infinite. *Journal of Combinatorial Theory, Series B*, 52(1):53–66, 1991. doi:10.1016/0095-8956(91)90090-7.

[45] R. J. Nowakowski and P. Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2-3):235–239, 1983. doi:10.1016/0012-365X(83)90160-7.

[46] A. Pnueli, A. Lempel, and S. Even. Transitive orientation of graphs and identification of permutation graphs. *Canadian Journal of Mathematics*, 23(1):160–175, 1971. doi:10.4153/CJM-1971-016-5.

[47] A. Quilliot. *Jeux et pointes fixes sur les graphes*. PhD thesis, Université de Paris VI, 1978.

[48] F. S. Roberts. Indifference graphs. In F. Harary, editor, *Proof Techniques in Graph Theory: Proceedings of the 2nd Ann Arbor Graph Theory Conference*, pages 139–146. Academic Press, 1969.

[49] L. Royer, M. Reimann, B. Andreopoulos, and M. Schroeder. Unraveling protein networks with power graph analysis. *PLoS Computational Biology*, 4(7):e1000108, 2008. doi:10.1371/journal.pcbi.1000108.

[50] K. Takamizawa, T. Nishizeki, and N. Saito. Linear-time computability of combinatorial problems on series-parallel graphs. *Journal of the ACM*, 29(3):623–641, 1982. doi:10.1145/322326.322328.

[51] W. T. Trotter. *Combinatorics and Partially Ordered Sets: Dimension Theory*, volume 6. The Johns Hopkins University Press, 2001.

[52] T. C. van Dijk, M. Fink, N. Fischer, F. Lipp, P. Markfelder, A. Ravsky, S. Suri, and A. Wolff. Block crossings in storyline visualizations. *Journal of Graph Algorithms and Applications*, 21(5):873–913, 2017. doi:10.7155/jgaa.00443.

[53] T. C. van Dijk, F. Lipp, P. Markfelder, and A. Wolff. Computing storyline visualizations with few block crossings. In F. Frati and K.-L. Ma, editors, *25th International Symposium on Graph Drawing and Network Visualization (GD'17)*, volume 10692 of *LNCS*, pages 365–378. Springer, 2017. doi:10.1007/978-3-319-73915-1_29.

[54] G. Wegner. *Eigenschaften der Nerven Homologisch-Einfacher Familien im* $\mathbb{R}^n$. PhD thesis, Universität Göttingen, 1967.

[55] C.-W. Yu and G.-H. Chen. Efficient parallel algorithms for doubly convex-bipartite graphs. *Theoretical Computer Science*, 147(1-2):249–265, 1995. doi:10.1016/0304-3975(94)00220-D.

[56] J. X.-S. Zheng, S. Pawar, and D. F. M. Goodman. Further Towards Unambiguous Edge Bundling: Investigating Power-Confluent Drawings for Network Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2019. doi:10.1109/TVCG.2019.2944619.

[57] H. Zhou, Panpan Xu, X. Yuan, and H. Qu. Edge bundling in information visualization. *Tsinghua Science and Technology*, 18(2):145–156, 2013. doi:10.1109/TST.2013.6509098.