Final Report of the Joint DFG–GAČR Project

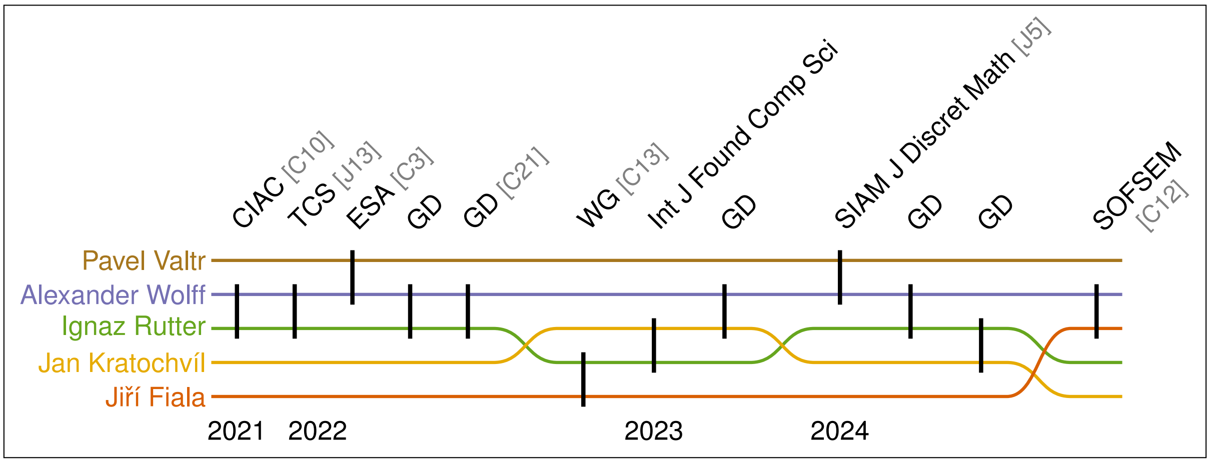
“Geometric Representation of Graphs”

Ignaz Rutter Alexander Wolff

June 24, 2026

1 General Information

DFG reference:	Ru 1903/3-1 and Wo 758/11-1
Project number:	412962361
Project title:	Geometric Representation of Graphs
Applicants:	Prof. Ignaz Rutter and Prof. Alexander Wolff
Affiliations:	I. Rutter, Chair of Theoretical Computer Science, Universität Passau A. Wolff, Chair of Algorithms & Complexity (Informatik I), Universität Würzburg
Co-applicant:	Dr. Jiří Fiala, Department of Applied Mathematics, Charles University Prague
Reporting period:	Passau: 01.10.2019 – 30.11.2023, Würzburg: 01.10.2019 – 30.06.2024 (including extensions)



The storyline above depicts inter-site publications during the funding project of the period. The three sites were Passau (Ignaz Rutter), Würzburg (Alexander Wolff), and Prague (Jiří Fiala with Jan Kratochvíl and Pavel Valtr). The publications that contributed to the project have keys (in gray) that refer to Section 4.

2 Summary

Important types of graph representations include *drawings* (where nodes are represented by points and edges by curves connecting their endpoints) and *intersection representations* (where nodes are represented by geometric objects whose intersection corresponds to an edge).

For dynamic graphs, one considers sequences of snapshots of a network and seeks stable representations in which unchanged parts of the network remain unchanged in the representation. This leads to two challenging problems: In the *representation extension problem*, a representation of a subgraph is given and must be extended to a complete representation; in the *simultaneous representation problem*, several graphs must be represented so that common parts are represented identically.

We investigated numerous variants of these problems. Particularly noteworthy are the proof of NP-completeness of the extension problem for circular arc graphs and an algorithm for the simultaneous representation of interval graphs for an arbitrary number of input graphs. Both results answer open questions that had been open for more than ten years. In the area of planar graph representations, we developed a new algorithm for synchronized planarity that reduces the running time for clustered planarity from $O(n^{16})$ to $O(n^2)$. Its implementation was integrated into the publicly available Open Graph Drawing Framework.

One can also use sequences of snapshots to decompose a graph into simple subgraphs and present them one after another, preventing viewers from being overwhelmed by the complexity of the entire graph. Such a sequence of snapshots is called a *storyplan*. Deciding whether a given graph admits a planar storyplan is NP-hard. We showed that the problem remains NP-hard if we insist on outerplanar or acyclic storyplans. For certain graph classes, we proved that all (or no) graphs in the class admit storyplans of a particular type.

A special case of intersection representations are *contact representations* of graphs, in which the interiors of the geometric objects are not allowed to intersect and edges are represented by contacts between the objects. We studied contact representations by convex polygons in three-dimensional space and showed that every graph admits such a representation when contacts occur at the vertices of the polygons. However, if contacts must occur along complete polygon edges, then dense graphs cannot be represented.

A completely different type of visualization represents graphs as *visibility graphs* of points. In this setting, obstacles are used to block lines of sight and thereby represent non-edges. The number of obstacles required to represent a graph in this way is called its *obstacle number*. This parameter has been studied extensively; we improved existing lower bounds and showed that the obstacle number is fixed-parameter tractable with respect to the vertex cover number of the given graph.

Zusammenfassung

Wichtige Darstellungen von Graphen sind *Zeichnungen* (wo Knoten als Punkte und Kanten als Kurven dargestellt werden) und *Schnittrepräsentationen* (wo Knoten durch geometrische Objekte repräsentiert werden, deren Schnitte Kanten entsprechen).

Für dynamische Graphen betrachtet man Folgen von Schnappschüssen eines Netzwerks und sucht stabile Repräsentationen, bei denen unveränderte Netzwerkteile auch in der Darstellung unverändert bleiben. Daraus ergeben sich zwei herausfordernde Probleme: beim Erweiterungsproblem ist die Darstellung eines Teilgraphen vorgegeben und zu ergänzen; beim simultanen Darstellungsproblem sollen mehrere Graphen so dargestellt werden, dass gemeinsame Teile identisch repräsentiert werden.

Wir haben zahlreiche Varianten dieser Probleme untersucht. Hervorzuheben sind der Nachweis der NP-Vollständigkeit des Erweiterungsproblems für Kreisbogengraphen sowie ein Algorithmus zur simultanen Darstellung von Intervallgraphen für beliebig viele Eingabegraphen. Beide Resultate beantworten mehr als zehn Jahre alte offene Fragen. Für planare Graphrepräsentationen wurde zudem ein neuer Algorithmus für *synchronized planarity* entwickelt, der die Laufzeit für *Clustered Planarity* von $O(n^{16})$ auf $O(n^2)$ reduziert. Die im DFG-Projekt entwickelte Implementierung wurde in das Open Graph Drawing Framework integriert und öffentlich verfügbar gemacht.

Man kann Folgen von Schnappschüssen auch dazu verwenden, um *einen* Graphen so in einfache Teilgraphen zu zerlegen und diese nacheinander zu zeigen, dass Betrachter nicht durch die Komplexität des ganzen Graphen überfordert werden. Eine solche Folge von Schnappschüssen heißt *Storyplan*. Es ist NP-schwer zu entscheiden, ob ein gegebener Graph einen planaren Storyplan zulässt. Wir haben gezeigt, dass die Frage auch für außenplanare oder kreisfreie Storyplans NP-schwer ist. Außerdem konnten wir für gewisse Graphklassen beweisen, dass alle (oder keine) Graphen der Klasse einen Storyplan eines bestimmten Typs zulassen.

Ein Spezialfall von Schnittrepräsentationen sind *Kontaktrepräsentationen* von Graphen, bei denen sich die Inneren der geometrischen Objekte nicht schneiden dürfen und Kanten durch Berührungen repräsentiert werden. Wir haben Kontaktrepräsentationen von konvexen Polygonen im dreidimensionalen Raum betrachtet und konnten zeigen, dass jeder Graph eine solche Repräsentation besitzt, wenn die Kontakte an Ecken der Polygone stattfinden. Falls man jedoch auf Kontakten entlang ganzer Polygonseiten besteht, können Graphen mit hoher Kantendichte nicht repräsentiert werden.

Eine ganz andere Art der Visualisierung repräsentiert Graphen als *Sichtbarkeitsgraphen* von Punkten. Dabei werden Hindernisse genutzt, um Sichtbarkeiten zu blockieren. Die Anzahl der Hindernisse, die man braucht, um einen Graphen so darzustellen, ist seine *Hinderniszahl*. Sie wurde schon intensiv studiert; wir konnten untere Schranken verbessern und zeigen, dass die Hinderniszahl festparameterberechenbar bezüglich der Knotenüberdeckungszahl des gegebenen Graphen ist.

3 Progress Report

Below we present our findings. For convenience we structure the results along the research directions (D1)–(D6) of the proposal. Further results that do not naturally fit into one of these directions are reported in a section titled “further research”. Some of our publications make contributions to more than one research direction. For example, it was often natural to study both the partial representation extension and the simultaneous representation problem together. For the purpose of this report, we mention such publications only once, for the research direction where we see its main contribution. Concerning the research of our collaborators from Czechia, we refer to the report they submitted to the Czech Science Foundation GAČR, which we attach to this report. In our report, we omit the research direction “Proximity and Low-Ply Drawings of Graphs” (D6), which was handled by our Czech colleagues.

In addition to this, we would like to point out some of our additional measures. For one, to ensure verifiability and validity of our findings, we have made an effort to integrate our implementations into the OGDF[1] where possible. In particular, OGDF now contains our implementation of PC-trees [J15] as well as our implementation of the synchronized planar algorithm [J16], which allows to efficiently solve the notorious c-planarity problem also in practice. For the former, we additionally obtained a badge from the ACM Journal of Experimental Algorithmics that certifies the reproducibility of our experimental results. Finally, we report that from November 6 to November 9 2023, we held a joint workshop at the University of Passau. The program contained a mix of presentations of late-breaking results, an open problem session as well as problem solving sessions that sparked follow-up work such as [I1], some of which is however not yet published.

3.1 Partial Representation Extension (D1)

In the partial representation extension problem, the input consists of a triplet $(G, H, \mathcal{H})$, where G is the input graph, $H \subseteq G$ is a subgraph and $\mathcal{H}$ is a representation of H (either an intersection representation or a drawing). The question is whether the representation $\mathcal{H}$ can be *extended* to a representation $\mathcal{G}$ of G without modifying the given parts of the representation, i.e., the representation of H induced by $\mathcal{G}$ must coincide with $\mathcal{H}$. We note that the problem depends significantly on the class of representations we consider. For a class $\mathcal{C}$ of graphs we usually consider the associated types of representations and denote the corresponding variant of the partial representation extension problem by RepExt($\mathcal{C}$). Notable classes of graphs that we worked on include permutation graphs (Perm), circular permutation graphs (CPerm), comparability graph (Comp), interval graphs (Int), circle graphs (Circle), and circular arc graphs (CA).

Comparability graphs and (circular) permutation graphs. Both RepExt(Perm) and Sim(Perm) have been studied and solved some time ago. As Perm is precisely the class of graphs that are both comparability and co-comparability graphs, it is not surprising that the solutions to both problems are strongly related to the problem of recognizing comparability graphs. In fact, Klavík et al. [32] solved RepExt(Comp), which asks whether a given orientation of a part of a graph G can be extended to a transitive orientation of G, in $O(\Delta m)$ time using an algorithm similar to Golumbic's algorithm for recognizing Comp [25]. They then extended this to an algorithm for RepExt(Comp) with the same running time. Similarly, Jampani and Lubiw [29] solved Sim(Perm) based on an algorithm for Sim(Comp), which asks whether a simultaneous graph admits a simultaneous transitive orientation, where shared edges are oriented the same. Their algorithm is based on Golumbic's algorithm and runs in $O(n^2)$ time. However, the fact that they have to apply it for the input graph and its complement, implies that they need $O(n^3)$ time to solve Sim(Perm) $O(n^3)$.

In our work presented at ISAAC'22 [C24, J23], which is partly based on the master thesis by Miriam Münch [40], who subsequently joined our group, we improved the running time of all these algorithms to optimal linear time. The key ingredient is the use of the modular decomposition, which compactly describes all possible transitive orientations of a graph by breaking the choice down into simpler independent choices, which are either binary or consist in choosing an arbitrary ordering for some subset of vertices [22]. Based on this, we were able to obtain a representation of all transitive orientation of a graph G that extend a given orientation of some subgraph $H \subseteq G$, thereby allowing us to solve RepExt(Comp) in linear time. Moreover, we constructed a representation of all transitive orientations of a graph H that extend to some transitive orientation of a supergraph $H \supseteq G$, expressed in terms of

[1] Open Graph Drawing Framework, `www.ogdf.net`

constraints on the modular decomposition of H. The fact that we can efficiently intersect such representations then allowed us to solve SIM(COMP) in linear time. While this already improves the running times for REPEXT(PERM) and SIM(PERM) to $O(n^2)$, we were further showed that the nodes of the modular decomposition can directly be associated with certain induced subgraphs (and their representations) of the input graph. This avoids the necessity to work with the complement and speeds up the algorithms to run in optimal linear time. Additionally, we gave the first solution for REPEXT and SIM on circular permutation graphs, which are defined in terms of segments between concentric circles. While REPEXT(CPERM) can be solved in linear time, our approach for SIM(CPERM) takes quadratic time.

Circular-arc graphs. Concerning the partial representation extension problem, we have considered several types of representations and determined the complexity of the problem for new graph classes or improved the running time of existing algorithms. One focus was the class CA of circular arc graphs, the last major graph class for which the recognition was still open. We managed to show that the problem is NP-complete in general [C13] and investigated several subclasses of CA. In particular, we could show that the partial representation extension problem is polynomial-time solvable for normal proper Helly representations, normal Helly representations, and even for Helly representations, as long as the predrawn vertices have distinct endpoints. Interestingly, the problem becomes NP-complete for Helly representations if we allow the predrawn vertices to share endpoints.

Circle graphs. Concerning circle graphs, we [C9, J11] followed up on previous work of Chaplick, Fulek and Klavík [12], who gave an $O(n^3)$-time algorithm for REPEXT(CIRCLE) and showed that SIM(CIRCLE) is NP-complete. We followed up on the question of Klavík et al., whether split trees can be used to improve the running time of the algorithm. To this end, we developed a way to canonically orient chord diagrams (with respect to a reference chord) and to show that the split tree developed by Gioan et al. [24, 23] can be used to compactly describe all the possible chord diagrams for a given circle graph. Similar to how the SPQR-tree of a biconnected graph represents all its planar embeddings and similar to how we employ the modular decomposition for comparability and permutation graphs above, it turns out with the canonical orientation, the choice of a (now canonically oriented) chord diagram corresponds bijectively to choosing canonically oriented chord diagrams for the skeleton graphs of the split decomposition tree. This data structure can be further extended to only represent the chord diagrams of a graph H that extend to a given supergraph G of H. To solve REPEXT(CIRCLE), it then suffices to construct this data structure and to test whether the resulting split tree represents the given representation of H. Altogether, this improves the running time of this approach to near-linear time $O((n+m)\alpha(n))$, matching the best known bound for the recognition problem.

In addition, it is worth mentioning that in subsequent work we managed to improve the running time of the recognition for circle graphs (and consequently also the running time for the partial representation extension problem) to optimal linear time [41]

We note that the use of our representation of all possible chord diagrams of a given graph is of independent interest and may well be usable to solve other constrained representation problems for circle graphs as they occur for example in Hsu's algorithm for recognizing circular-arc graphs. We also showed that it can be further extended to solve SIM(CIRCLE) in $O(n^6)$ time if the shared graph is connected. The result is currently only contained in the PhD thesis of Peter Stumpf [48], as we still hope to substantially improve the running time.

Rectangular duals. A rectangular dual of a graph G is a contact representation of G by axis-aligned rectangles such that (i) no four rectangles share a point and (ii) the union of all rectangles is a rectangle. Combinatorially, a rectangular dual can be described by a regular edge labeling (REL), which determines the orientations of the rectangle contacts.

It has long been known that a plane internally-triangulated graph has a representation with only four rectangles touching the outer face if and only if its outer face is a 4-cycle and it has no separating triangles, that is, a triangle whose removal disconnects the graph [38]. Such a graph is called a *properly-triangulated planar (PTP)* graph. Kant and He [31] have shown that a rectangular dual of a given PTP graph can be computed in linear time.

We [C10, J13] described linear-time algorithms for the partial representation extension problem and the simultaneous representation problem for rectangular duals when each input graph is given together with a REL. Both algorithms are based on formulations as linear programs, yet they have geometric interpretations and can be seen as extensions of the above-mentioned algorithm of Kant and He [31].

Planarity and directed planarity variants. While the seminal result of Angelini et al. [2] solves the partial representation extension problem for planar drawings in optimal linear time, the algorithm is rather complicated as it is based on multiple types of decompositions of the input graph, particularly to deal with the case where the partial drawing is disconnected. We developed a simpler algorithm that follows the standard vertex-addition paradigm for planarity testing, where the major difference is that enhance the underlying data structure, a PC-tree, so that it handles the constraints mostly transparently. Our article [C18] appeared in SOSA, the symposium on simplicity in algorithms. In this context, we also consider a related question. Namely, which planar graphs admit a planar drawing extension, for an arbitrary drawing of its outer face as a simple polygon? We call such graphs *polygon-universal* and showed that they can be characterized by two simple forbidden substructures [C25, J24]. We also studied the drawing extension problem for orthogonal drawings, where we showed that minimizing the number of bends or testing whether two bends per edge suffice are NP-complete, yet the number of bends per edge can be bounded in terms of the size of the predrawn part [3, J2].

We further considered variants of planarity that are particularly designed for directed graphs, namely upward planarity and level planarity. Here the main issue to overcome is that the recognition problem is NP-hard for upward planar graphs, and while level planarity can be tested in linear time [30], the algorithm is very complicated, and the structure of the different kinds of drawings is poorly understood. We therefore started a more systematic investigation of more restricted problem variants. Here we studied the interplay between several directed planarity variants, such as (radial) upward/level planarity testing with a fixed embedding, with a fixed upward embedding, or even with a fixed level embedding [J10].

Additionally, we investigated the option to extend SPQR-tree-based approaches to upward and level planarity. To this end, we proved that every biconnected single-source graphs G admit a SPQR-tree-like decomposition tree $\mathcal{T}$, called UP-tree, that combinatorially describe all upward-planar embeddings of the graph [C7]. The embedding choice is thereby decomposed into independently (i) arbitrarily permuting parallel subgraphs between two pole vertices and (ii) mirroring certain components whose upward-planar embedding is unique up to reflection. By using the UP-tree as a drop-in replacement for the SPQR-tree, we can therefore transfer algorithms from the planar setting to the upward-planar setting without modification. We were able to prove a similar result for level-planar embeddings of biconnected single-source level graphs [C8, J9], which enabled us to solve RepExt(LevelPlanar) in linear time for biconnected single-source level graphs in linear time, thereby improving the running time over an earlier algorithm running in time $O(n^2)$ [8] as well as Sim(LevelPlanar) for instances whose shared graph is biconnected and has a single-source level graph, which was previously open.

3.2 Simultaneous Representations (D2)

The simultaneous representation problem has several input graphs $\{G_i\}_{i=1}^k$, which may share vertices and edges. The question is whether there exist corresponding representations $\{\mathcal{R}_i\}_{i=1}^k$ such that for $i, j \in [k]$ we have that $\mathcal{R}_i$ and $\mathcal{R}_j$ coincide on the graph $G_i \cap G_j = (V(G_i) \cap V(G_j), E(G_i) \cap E(G_j))$. Since this problem is NP-complete in essentially all cases, we restrict our attention to the sunflower case, which additionally requires that there is a shared graph S such that $G_i \cap G_j = S$ for each $i \neq j \in [k]$. That is, any two distinct input graphs share exactly the same shared subgraph S. For a class $\mathcal{C}$ of graphs with corresponding representations, we denote this problem by Sim($\mathcal{C}$). The problem is equivalent to asking whether S admits a representation that can be extended to a representation of each input graph. In some cases, this allows to reduce RepExt($\mathcal{C}$) to Sim($\mathcal{C}$).

Proper and unit interval graphs. The partial representation extension problem for proper and unit interval graphs was studied and resolved as early as 2014 [33]. We studied the problem of recognizing simultaneous proper and unit interval graphs [C28, J27]. For simultaneous proper interval graphs, we proved a characterization based on simultaneous straight enumerations, thereby mimicking a classical characterization of proper interval graphs [43]. As in the case of the recognition problem, this characterization can be used to straightforwardly recognize simultaneous proper interval graphs in linear time using a PQ-tree-based algorithm. The case of unit intervals, however, is much more tricky due to the geometric nature of the problem. While every simultaneous unit interval representation yields a simultaneous straight enumeration, the converse is not true. The key ingredients of our algorithm with running time $O(|V(\bigcup \mathcal{G})| \cdot |E(\bigcup \mathcal{G})|)$ are (i) a characterization of the simultaneous straight enumerations that can be realized with unit intervals and (ii) a compact description of all simultaneous straight enumerations of a simultaneous graph that allows to efficiently search one that can be realized with unit interval graphs.

Interval graphs. Interval graphs were the first geometric intersection graphs for which REPEXTand SIMwere investigated. The partial representation extension problem was quickly solved in polynomial [35] and later even linear time [34, 7]. Jampani and Lubiw, who introduced the idea of simultaneous intersection representations, gave an $O(n^2 \log n)$-time algorithm for $\text{SIM}^{k=2}(\text{INT})$ and asked whether the problem can also be solved for $k > 2$. Bläsius and Rutter [7] gave a linear-time algorithm, but the case of $k > 2$ remained open. In our work presented at ESA'23 [C29], we gave an algorithm with running time $O(|V(\bigcup \mathcal{G})| + |E(\bigcup \mathcal{G})|)$, thereby answering this open question more than ten years later.

Constrained planarity. Finally, spawned by the breakthrough result of Fulek and Tóth [20, 21], who gave the first polynomial-time algorithm for the famous problem CLUSTERED PLANARITY, right around the start of our project, we considered clusterings of graphs as a source of constraints. A *clustering* of a graph $G = (V, E)$ is a rooted tree T whose leaf set is V. Each node μ of T defines a *cluster* C_μ that contains those vertices in V that lie in the subtree rooted at T. The problem CLUSTERED PLANARITY asks whether a given clustered graph admits a clustered-planar drawing, which is a planar drawing of the graph where for each cluster C_μ, we can draw a simply connected region R_μ such that (i) the boundaries of the regions are pairwise disjoint, (ii) each region R_μ encloses exactly the vertices in cluster C_μ and (iii) each edge crosses each cluster boundary at most once. In particular, condition (iii) implies that an edge may cross the boundary of a cluster if and only if one of its endpoints belongs to the cluster and the other one does not. Inspired by the work of Fulek and Tóth [20, 21], whose algorithm for CLUSTERED PLANARITY runs in time $O(n^{16})$, and building on earlier work by Bläsius and Rutter [7], we gave a new algorithm that is much simpler and faster; it runs in quadratic time [C6, J8].

On a high level, the algorithm works several iterations, each of which consists in choosing a suitable constraint and applying one of three operations on it. As there are $O(m)$ iterations and each operation can be executed in time $O(m)$, the overall running time is $O(m^2)$. As it turns out, the operations can be executed faster, provided the PQ-tree defining the possible rotations around a certain vertex are already computed. To achieve this, we [C17, J17] designed a designated data structure that allows to maintain the SPQR-tree under suitable operations and thus to provide the required PQ-trees faster. This allows us to improve the running time of the algorithm to $O(m \cdot \Delta)$, where Δ is the maximum degree of the instance. For CLUSTERED PLANARITY this translates to a running time of $O(dn)$, where d is the maximum number of edges that crosses a cluster boundary, which is significantly faster unless there is a quadratic number edges that cross cluster boundaries. On a high level, the algorithm works several iterations, each of which consists in choosing a suitable constraint and applying one of three operations on it. As there are $O(m)$ iterations and each operation can be executed in time $O(m)$, the overall running time is $O(m^2)$. As it turns out, the operations can be executed faster, provided the PQ-tree defining the possible rotations around a certain vertex are already computed. By designing a designated data structure that allows to maintain the SPQR-tree under suitable iterations. This allows us to improve the running time of the algorithm to $O(m \cdot \Delta)$, where Δ is the maximum degree of the instance. For CLUSTERED PLANARITY this translates to a running time of $O(dn)$, where d is the maximum number of edges that crosses a cluster boundary. This if often significantly faster than $O(n^2)$.

Our attempts at implementing the SYNCHRONIZED PLANARITY algorithm were first hindered by the fact that there is no working implementation of PC-trees available. In our work [C14, J15], which won the best paper award in ESA'19, together with our Bachelor student Matthias Pfretzschner, who in the meanwhile joined our group, we developed two different implementations of PC-trees. One that follows (and fixes several mistakes in) the original description of PC-trees by Hsu and McConnell [27, 28] and a second one that employs a union find data structure. Though the second one has a worse time-complexity, it is twice as fast as the other one and four times faster than the fastest fully correct competing implementation of a PQ-tree data structure.

Based on this, we were recently able to present a working implementation of SYNCHRONIZED PLANARITY along with several improvements that ensure that it runs fast in practice [C16, J16]. Both the PC-tree implementation and the SYNCHRONIZED PLANARITY implementation have been integrated into the OGDF library[2] and are now publicly available and can be used to solve a large variety of constrained planar embedding problems, including the notorious c-planarity problem.

Finally, we [C15] recently began the investigation of the parameterized complexity of SIM(PLANAR). Namely, we showed that the problem is intractable with respect to several natural parameters of the shared graph, such as the maximum degree and the vertex cover number but is FPT with respect to their sum.

[2]OGDF

3.3 Visibility Representations with Obstacles (D3)

An *obstacle representation* of a graph G consists of a set of pairwise disjoint simply-connected closed regions and a one-to-one mapping of the vertices of G to points such that two vertices are adjacent in G if and only if the line segment connecting the two corresponding points does not intersect any obstacle. The *(convex) obstacle number* of a graph is the smallest number of (convex) obstacles in an obstacle representation of the graph in the plane such that all obstacles are simple polygons. Alpert, Koch, and Laison [1] introduced the obstacle number and the convex obstacle number. Using Ramsey theory, they proved that the convex obstacle number (and hence the obstacle number) is unbounded.

It was known that the obstacle number of each n-vertex graph is $O(n \log n)$ [5] and that there are n-vertex graphs whose obstacle number is $\Omega(n/(\log\log n)^2)$ [16]. We [C3, J5] improved this lower bound to $\Omega(n/\log\log n)$ for simple polygons and even to $\Omega(n)$ for the special case of convex polygons. To obtain these stronger bounds, we improved known estimates on the number of n-vertex graphs with bounded obstacle number, solving a conjecture of Dujmović and Morin [16].

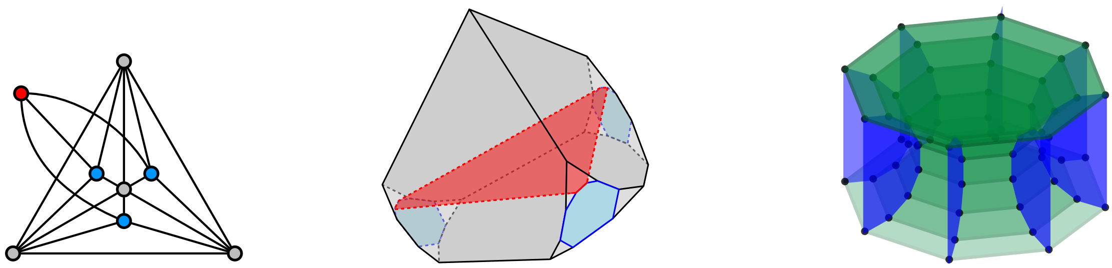
We complement these combinatorial bounds by two complexity results. First, we show that computing the obstacle number of a graph G is fixed-parameter tractable in the vertex cover number of G. Second, we show that, given a graph G and a simple polygon P, it is NP-hard to decide whether G admits an obstacle representation using P as the only obstacle. Interestingly, the complexity of deciding whether a given graph has obstacle number 1 is still open.

3.4 (Topological and Geometric) Intersection Representations (D4)

We note that several of our works that are concerned with geometric intersection representations deal with the particular aspects of partial representation extension and simultaneous representations and we refer to Section 3.1 and Section 3.2. Here we focus on results that do not relate to these problems.

Side contacts of convex polygons in 3D. We [C1, J3] studied whether a given graph can be realized as an adjacency graph of the polygonal cells of a polyhedral surface in $\mathbb{R}^3$. We showed that every graph is realizable as a polyhedral surface with arbitrary polygonal cells, and that this is not true if we require the cells to be convex. In particular, if the given graph contains K_5, $K_{5,81}$, or any nonplanar 3-tree as a subgraph, no such realization exists. On the other hand, all planar graphs, $K_{4,4}$, and $K_{3,5}$ can be realized with convex cells. The same holds for any subdivision of any graph where each edge is subdivided at least once, and, by a result from McMullen, Schulz, and Wills [39], for any hypercube.

Our results for "side contacts" between polygons are in contrast to our earlier work on "corner contacts", which allowed us to represent *every* graph as contact graph of convex polygons in 3D [C11] (albeit using grids of exponential size). For a corner-contact representation of $K_{8,8}$, see Fig. 1c.



(a) A nonplanar graph G. (b) A side-contact representation of G. (c) A corner-contact representation of $K_{8,8}$.

Figure 1: Side- and corner-contact representations (of two different graphs).

Our results have implications on the maximum density of graphs describing polyhedral surfaces with convex cells: The realizability of hypercubes shows that the maximum number of edges over all realizable n-vertex graphs is in $\Omega(n \log n)$. From the non-realizability of $K_{5,81}$, we obtain via the Kővari–Sós–Turán Theorem [37] that every realizable n-vertex graph has $O(n^{9/5})$ edges. As such, these graphs can be considerably denser than planar graphs, but not arbitrarily dense. This upper bound has later been improved to $O(n^{5/3})$ by Schulz [47]. To this end, he showed that $K_{3,250}$ does not admit a side-contact representation.

Stick graphs. Stick graphs are intersection graphs of horizontal and vertical line segments that all touch a line of slope -1 and lie above this line. There are three variants of stick graph recognition: (V1) when nothing further is given, (V2) when an ordering of the horizontal sticks is given, and (V3) when an ordering of both types of sticks is given. It was known how to solve (V3) efficiently [15]. We [J14] gave a faster algorithm for (V3) and the first efficient solution for (V2). Later, it turned out that (V1) is NP-complete [44]. We also considered a research direction suggested by Cabello and Jejčič [9], namely stick graphs where, for each vertex, the length of its stick is prescribed. The case that the sticks are required to be of equal length corresponds to (special) bipartite permutation graphs, which can be recognized in linear time. However, we showed that, in the presence of arbitrary length restrictions, all three before-mentioned recognition problems are NP-complete.

Planar L-drawings of directed graphs. For drawing directed graphs, one can use the *L-drawing standard*, where each edge is represented by an "L", that is, a vertical line segment incident to the source of the edge and a horizontal line segment incident to the target. We [10] considered planar L-drawings (where no two Ls are allowed to cross; see Figs. 2c and 2d) and the special case of upward planar L-drawings (see Figs. 2a and 2b), where additionally every directed edge must point upwards. Since edges in planar L-drawings are allowed to overlap, we draw the edges as "rounded" Ls to clarify which pairs of vertices are actually connected.

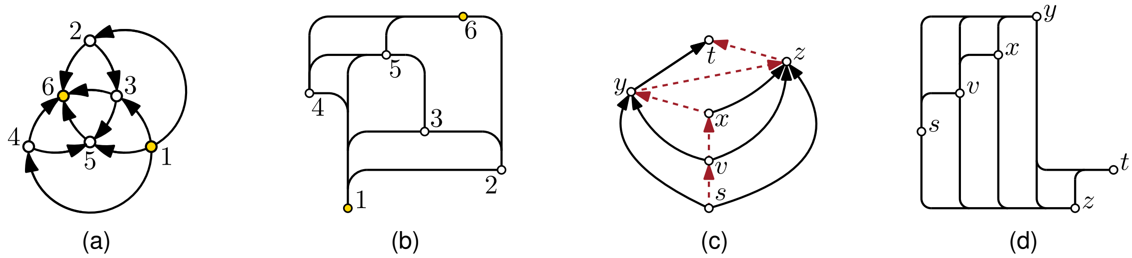


Figure 2: (a) A bitonic st-orientation of the octahedron that admits an upward-planar L-drawing (b). (c) An upward-planar st-graph that does not admit an upward-planar L-drawing. (d) A planar L-drawing of the graph in (c). Note that the edges (x, z), (y, z), and (y, t) are not drawn upward.

We first proved that testing whether a given directed graph admits a planar L-drawing is NP-complete. On the other hand, we showed how to decide in linear time whether there exists a planar L-drawing of a plane directed graph with a fixed assignment of the edges to the four sides (top, bottom, left, and right) of the vertices.

Then we provided upper bounds on the maximum number of edges of graphs that admit upward-planar L-drawings. An st-graph is a directed planar graph with a single source and a single sink that lie on the same face. We characterized st-graphs admitting an upward-planar L-drawing as exactly those st-graphs admitting an embedding that supports a bitonic st-ordering (for an example, see Fig. 2a).

The complexity of finding tangles. Given a finite set of y-monotone Jordan curves, called *wires*, a *tangle* determines the order of the wires on a number of horizontal *layers* such that the orders of the wires on any two consecutive layers differ only in swaps of neighboring wires. Given a multiset L of *swaps* (that is, unordered pairs of wires) and an initial order of the wires, a tangle *realizes* L if each pair of wires changes its order exactly as many times as specified by L. For example, the tangles T and T' in Fig. 3 both realize the list $L = \{(1,2),(1,3),(1,4),(2,3)\}$ using $\mathrm{id}_4 = \langle 1,2,3,4\rangle$ as the start permutation. The order changes of the wires result in crossings, hence a tangle is a drawing of an edge-weighted intersection graph whose vertices correspond to the wires.

ListFeasibility is the problem of finding a tangle that realizes a given list L for a prescribed initial order if such a tangle exists. The problem TangleMinimization additionally insists that the desired tangle uses the minimum number of layers. For example, the tangle T' in Fig. 3b is a solution of TangleMinimization for the list L and the start permutation id_4. ListFeasibility (and therefore TangleMinimization) is NP-hard [50]. Tangles are visualizations of *chaotic attractors*, which occur in dynamic systems. Such systems are considered in various areas of the natural sciences [14].

We [C19, J18] proved that ListFeasibility remains NP-hard if every pair of wires swaps only a constant number of times. On the positive side, we present an algorithm for TangleMinimization that

(a) (b)

Figure 3: Tangles T and T' of different heights realizing the list $L = \{(1, 2), (1, 3), (1, 4), (2, 3)\}$.

computes an optimal tangle for n wires and a given list L of swaps in $O((|L|/n^2 + 1)^{n^2/2} \cdot \varphi^n \cdot n! \cdot n \cdot \min\{|L|, n^2\} \cdot \log |L|)$ time, where $\varphi \approx 1.618$ is the golden ratio and $|L|$ is the total number of swaps in L.

Obviously, this algorithm solves ListFeasibility, too. We showed that ListFeasibility can also be solved by a simpler and faster version of the algorithm. Moreover, the algorithm helped us to show that ListFeasibility is in NP and fixed-parameter tractable with respect to the number of wires. For *simple* lists, where every swap occurs at most once (as in the list L mentioned above), we showed how to solve TangleMinimization in $O(n!\varphi^n)$ time. Interestingly, for this special case, whose complexity is still open, there exists a simple quadratic-time algorithm (odd–even sort, a parallel variant of bubblesort) that produces tangles with at most one layer more than the minimum [46].

3.5 Crossing Numbers (D5)

For computing the minimum number of crossings in straight-line drawings, we considered both theoretical and practical approaches.

From a theoretical point of view, we studied the problem geometric edge insertion [J26], which for a given graph G and an additional edge e asks for a straight-line drawing of $G + e$, where the subdrawing induced by G is crossing-free and the number of crossings is minimized subject to this condition. Here we gave polynomial-time algorithms for graphs of maximum degree 5 as well as FPT and approximation algorithms.

For topological drawings, iterated edge insertion is a common heuristic for crossing minimization in topological drawings. We developed heuristics that mimic this idea to minimize crossings in straight-line drawings [J25] that work substantially better than previous force-based methods. A limiting factor is the exact computation of a suitable arrangement, which we managed to improve by (i) limiting the size of the arrangement by employing primal and dual sampling techniques, based on the fact that the relevant range space has bounded VC-dimension, and (ii) working with a combinatorial representation of the arrangment, where exact computations are employed only when they are absolutely necessary [C26].

In follow-up work, we devised an XP algorithm (w.r.t. the natural parameter) for computing the *local circular crossing number* of a given graph, which means that, for any constant k, we can recognize outer k-planar graphs efficiently [36]. This answered in the affirmative a question that had been open since linear-time recognition algorithms for outer 1-planar graphs have been known [26, 4]. Our approach is based on structural insights that we gained earlier when giving better bounds for the treewidth of outer k-planar graphs [18]. Our XP algorithm improves upon our earlier pseudopolynomial-time algorithm [13].

3.6 Further Topics

Forest and outerplanar storyplans. With our Czech project partner, we studied the problem of gradually representing a complex graph as a sequence of drawings of small subgraphs whose union is the complex graph. The sequence of drawings is called *storyplan*, and each drawing in the sequence is called a *frame*. In an (outer)planar storyplan, every frame is (outer)planar; in a forest storyplan, every frame is acyclic. Figure 4 shows a forest storyplan for the Petersen graph. Binucci, Di Giacomo, Lenhart, Liotta, Montecchiani, Nöllenburg, and Symvonis [6] proved that every graph of treewidth at most 3 admits a planar storyplan and that deciding whether a given graph admits a *planar* storyplan is NP-complete. They also presented two FPT algorithms, one parameterized with respect to the vertex cover number and one with respect to the feedback edge set number of the input graph.

We [C12] proved that deciding whether a given graph admits an outerplanar storyplan (or a forest storyplan) is NP-complete. Then, we showed that the FPT algorithms of Binucci et al. also work for our problem variants with small modifications. Finally, we identified graph families that admit outerplanar

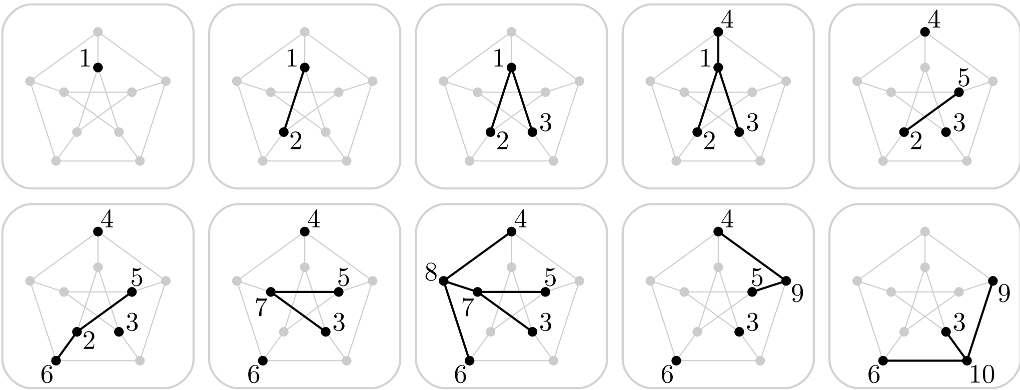


Figure 4: A forest storyplan for the Petersen graph.

and forest storyplans and graph families for which such storyplans do not always exist. In the affirmative case, we presented efficient algorithms that produce straight-line storyplans (as in Fig. 4).

Coloring problems. We also worked on coloring problems, which are a classical topic for geometric intersection graphs.

For circle graphs, the 3-coloring problem plays a special role. Unger [49] claimed a polynomial-time algorithm but many details were omitted and a journal version has not appeared. We [C2, J4] identified two crucial mistakes in Unger's work and thereby clarified the status of the 3-coloring problem for circle graphs as open. This has since sparked renewed interest into the problem as documented by a recent quasi-polynomial algorithm [45].

Motivated by an application in graph drawing [51], we also considered a coloring problem on mixed interval graphs [C21, C20]. A *mixed graph* is a graph that has undirected edges and directed arcs. When coloring the vertices of a mixed graph (with integers), one insists (as usual) that the endpoints of an edge receive different colors. Additionally, for each arc, the target vertex must receive a larger color than the start target. We considered various types of mixed interval graphs, where the definition of the arcs depends of the relative position of the intervals. For *directional interval graphs*, where there is an arc (u, v) if interval u overlaps and starts to the left of interval v, we [C21] presented an efficient algorithm that computes a coloring with the smallest number of colors. We also showed how to recognize directional interval graphs in quadratic time. For *containment interval graphs*, where there is an arc (u, v) if interval u contains interval v, we [C20] proved that it is NP-hard to decide whether a given graph can be colored with a given number of colors, but we presented an efficient algorithm that colors such a graph G with at most $2\omega(G) - 1$ colors, where $\omega(G)$ is the size of a largest clique in G. Hence, our algorithm is a 2-approximation algorithm.

Constrained planarity. We studied constrained planarity variants that are derived from the NodeTrix drawing style, where some clusters of vertices are represented by adjacency matrices and edges between such clusters connect to the rows/columns of these matrices. In the case of clusters of size 2, we [J19] improved an earlier algorithm with running time $O(n^3)$. Based on this, we studied more general versions, where the rows and columns of the adjacency matrix can be permuted independently [C23, J21] and the parameterized complexity of a version where each vertex v is equipped with a set of PQ-trees $D(v)$, one of which must be satisfied by the rotation of v [C22, J20]. We also investigated a constrained version of planarity where a sliding window traverses a stream of edges and edges that appear in a common window may not cross [J22]. Of a rather geometric flavor in this context are our works on stretchability of planar drawings where additionally some curves are given that traverse the drawing and the task is to stretch the drawing in such a way that the curves becomes a line [C27] as well as our work on untangling planar drawings, where we showed that untangling a circular drawing, i.e., turning a straight-line drawing where all vertices lie on a circle into a planar circular drawing with the minimum number of vertex moves, is NP-complete but admits an approximation [C5, J7]. Finally, we also the problem of compaction in planar orthogonal drawings, which is known to be NP-complete but can be solved efficiently for so-called turn-regular representations. We [C4, J6] showed that these include all biconnected planar graphs of maximum degree 3 as well as all planar Hamiltonian graphs of maximum degree 4 and further characterized the trees that admit such a representation. Moreover, if a planar embedding is given and its faces are small, the existence of a turn-regular orthogonal representation can be determined efficiently.

Level planarity. Our work [J1] generalizes the notion of level planarity to the cases of standing and rolling cylinders as well as to the torus. While testing remains polynomial-time solvable, the simultaneous drawing problem becomes NP-complete on the torus even for (i) two graphs and three layers or (ii) three graphs on two layers. We further researched a characterization of Randerath et al. [42] for level planarity whose correctness was not entirely clear. Indeed, in the meanwhile we have demonstrated that the accompanying algorithm and thus the proof of Radnerath et al. is flawed [17]. In our work [J12] we show that the characterization of Randerath et al. is equivalent to the Hanani-Tutte theorem for level planarity [19] and thus is correct, and we further generalized this correspondence to the case of radial level-planar drawings, thereby obtaining a new test for radial level planarity. Motivated by the success of our work on synchronized planarity, see (D2), we also initiated the study of variants of clustered planarity for level graphs [I2].

References

[1] Hannah Alpert, Christina Koch, and Joshua D. Laison. Obstacle numbers of graphs. *Discrete Comput. Geom.*, 44(1):223–244, 2010. DOI: 10.1007/s00454-009-9233-8.

[2] Patrizio Angelini, Giuseppe Di Battista, Fabrizio Frati, Vít Jelínek, Jan Kratochvíl, Maurizio Patrignani, and Ignaz Rutter. Testing planarity of partially embedded graphs. *ACM Trans. Algorithms*, 11(4):32:1–32:42, 2015. DOI: 10.1145/2629341.

[3] Patrizio Angelini, Ignaz Rutter, and Sandhya T. P. Extending partial orthogonal drawings. In David Auber and Pavel Valtr, editors, *Proc. 28th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 12590 of *LNCS*, pages 265–278. Springer, 2020. DOI: 10.1007/978-3-030-68766-3_21.

[4] Christopher Auer, Christian Bachmaier, Franz J. Brandenburg, Andreas Gleißner, Kathrin Hanauer, Daniel Neuwirth, and Josef Reislhuber. Outer 1-planar graphs. *Algorithmica*, 74(4):1293–1320, 2016. DOI: 10.1007/S00453-015-0002-1.

[5] Martin Balko, Josef Cibulka, and Pavel Valtr. Drawing graphs using a small number of obstacles. *Discrete Comput. Geom.*, 59(1):143–164, 2018. DOI: 10.1007/s00454-017-9919-2.

[6] Carla Binucci, Emilio Di Giacomo, William J. Lenhart, Giuseppe Liotta, Fabrizio Montecchiani, Martin Nöllenburg, and Antonios Symvonis. On the complexity of the storyplan problem. *J. Comput. Syst. Sci.*, 139:103466:1–17, 2024. DOI: 10.1016/J.JCSS.2023.103466.

[7] Thomas Bläsius and Ignaz Rutter. Simultaneous pq-ordering with applications to constrained embedding problems. *ACM Trans. Algorithms*, 12(2):16:1–16:46, 2016. DOI: 10.1145/2738054.

[8] Guido Brückner and Ignaz Rutter. Partial and constrained level planarity. In Philip N. Klein, editor, *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2000–2011. SIAM, 2017. DOI: 10.1137/1.9781611974782.130.

[9] Sergio Cabello and Miha Jejčič. Refining the hierarchies of classes of geometric intersection graphs. *Electr. J. Combin.*, 24(1):P1.33, 2017. URL: http://www.combinatorics.org/ojs/index.php/eljc/article/view/v24i1p33.

[10] Steven Chaplick, Markus Chimani, Sabine Cornelsen, Giordano Da Lozzo, Martin Nöllenburg, Maurizio Patrignani, Ioannis G. Tollis, and Alexander Wolff. Planar L-drawings of directed graphs. *Comput. Geom. Topol.*, 2(1):7:1–7:15, 2023. DOI: 10.57717/cgt.v2i1.43.

[11] Steven Chaplick, Stefan Felsner, Philipp Kindermann, Jonathan Klawitter, Ignaz Rutter, and Alexander Wolff. Simple algorithms for partial and simultaneous rectangular duals with given contact orientations. *Theoret. Comput. Sci.*, 919:66–74, 2022. DOI: 10.1016/j.tcs.2022.03.031.

[12] Steven Chaplick, Radoslav Fulek, and Pavel Klavík. Extending partial representations of circle graphs. *J. Graph Theory*, 91(4):365–394, 2019. DOI: 10.1002/JGT.22436.

[13] Steven Chaplick, Myroslav Kryven, Giuseppe Liotta, Andre Löffler, and Alexander Wolff. Beyond outerplanarity. In Fabrizio Frati and Kwan-Liu Ma, editors, *Proc. 25th Int. Symp. Graph Drawing & Network Vis. (GD)*, volume 10692 of *LNCS*, pages 546–559. Springer, 2018. DOI: 10.1007/978-3-319-73915-1_42.

[14] James P. Crutchfield, J. Doyne Farmer, Norman H. Packard, and Robert S. Shaw. Chaos. *Scientific American*, 254(12):46–57, 1986. DOI: 10.1038/scientificamerican1286-46.

[15] Felice De Luca, Md. Iqbal Hossain, Stephen G. Kobourov, Anna Lubiw, and Debajyoti Mondal. Recognition and drawing of stick graphs. *Theor. Comput. Sci.*, 796:22–33, 2019. DOI: 10.1016/j.tcs.2019.08.018.

[16] Vida Dujmović and Pat Morin. On obstacle numbers. *Electr. J. Comb.*, 22(3):#P3.1, 2015. DOI: 10.37236/4373.

[17] Simon D. Fink, Matthias Pfretzschner, Ignaz Rutter, and Peter Stumpf. Level planarity is more difficult than we thought (poster abstract). In Stefan Felsner and Karsten Klein, editors, *Proc. 32nd International Symposium on Graph Drawing and Network Visualization (GD)*, LIPIcs, 50:1–50:3. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. DOI: 10.4230/LIPICS.GD.2024.50.

[18] Oksana Firman, Grzegorz Gutowski, Myroslav Kryven, Yuto Okada, and Alexander Wolff. Bounding the treewidth of outer k-planar graphs via triangulations. In Stefan Felsner and Karsten Klein, editors, *Proc. 32nd Int. Symp. Graph Drawing & Network Vis. (GD)*, volume 320 of *LIPIcs*, 14:1–17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. DOI: 10.4230/LIPIcs.GD.2024.14.

[19] Radoslav Fulek, Michael J. Pelsmajer, Marcus Schaefer, and Daniel Štefankovič. *Hanani–tutte, monotone drawings, and level-planarity*. In *Thirty Essays on Geometric Graph Theory*. János Pach, editor. Springer New York, New York, NY, 2013, pages 263–287. DOI: 10.1007/978-1-4614-0110-0_14.

[20] Radoslav Fulek and Csaba D. Tóth. Atomic embeddability, clustered planarity, and thickenability. In Shuchi Chawla, editor, *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2876–2895, 2020. DOI: 10.1137/1.9781611975994.175.

[21] Radoslav Fulek and Csaba D. Tóth. Atomic embeddability, clustered planarity, and thickenability. *J. ACM*, 69(2):13:1–13:34, 2022. DOI: 10.1145/3502264.

[22] Tibor Gallai. Transitiv orientierbare graphen. *Acta Math. Acad. Sci. Hungar.*, 18:25–66, 1967. DOI: 10.1007/BF02020961.

[23] Emeric Gioan, Christophe Paul, Marc Tedder, and Derek G. Corneil. Practical and efficient circle graph recognition. *Algorithmica*, 69(4):759–788, 2014. DOI: 10.1007/S00453-013-9745-8.

[24] Emeric Gioan, Christophe Paul, Marc Tedder, and Derek G. Corneil. Practical and efficient split decomposition via graph-labelled trees. *Algorithmica*, 69(4):789–843, 2014. DOI: 10.1007/S00453-013-9752-9.

[25] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

[26] Seok-Hee Hong, Peter Eades, Naoki Katoh, Giuseppe Liotta, Pascal Schweitzer, and Yusuke Suzuki. A linear-time algorithm for testing outer-1-planarity. *Algorithmica*, 72(4):1033–1054, 2015. DOI: 10.1007/S00453-014-9890-8.

[27] Wen-Lian Hsu and Ross M. McConnell. PC trees and circular-ones arrangements. *Theor. Comput. Sci.*, 296(1):99–116, 2003. DOI: 10.1016/S0304-3975(02)00435-8.

[28] Wen-Lian Hsu and Ross M. McConnell. PQ trees, PC trees, and planar graphs. In Dinesh P. Mehta and Sartaj Sahni, editors, *Handbook of Data Structures and Applications*. Chapman and Hall/CRC, 2004. DOI: 10.1201/9781420035179.CH32.

[29] Krishnam Raju Jampani and Anna Lubiw. The simultaneous representation problem for chordal, comparability and permutation graphs. *J. Graph Algorithms Appl.*, 16(2):283–315, 2012. DOI: 10.7155/JGAA.00259.

[30] Michael Jünger and Sebastian Leipert. Level planar embedding in linear time. *J. Graph Algorithms Appl.*, 6(1):67–113, 2002. DOI: 10.7155/JGAA.00045.

[31] Goos Kant and Xin He. Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems. *Theoret. Comput. Sci.*, 172(1):175–193, 1997. DOI: 10.1016/S0304-3975(95)00257-X.

[32] Pavel Klavík, Jan Kratochvíl, Tomasz Krawczyk, and Bartosz Walczak. Extending partial representations of function graphs and permutation graphs. In Leah Epstein and Paolo Ferragina, editors, *Proc. 20th Annual European Symposium on Algorithms (ESA)*, volume 7501 of *LNCS*, pages 671–682. Springer, 2012. DOI: 10.1007/978-3-642-33090-2_58.

[33] Pavel Klavík, Jan Kratochvíl, Yota Otachi, Ignaz Rutter, Toshiki Saitoh, Maria Saumell, and Tomás Vyskocil. Extending partial representations of proper and unit interval graphs. In R. Ravi and Inge Li Gørtz, editors, *Proc. 14th Scandinavian Symposium on Algorithm Theory (SWAT)*, volume 8503 of *LNCS*, pages 253–264. Springer, 2014. DOI: 10.1007/978-3-319-08404-6_22.

[34] Pavel Klavík, Jan Kratochvíl, Yota Otachi, Toshiki Saitoh, and Tomás Vyskočil. Extending partial representations of interval graphs. *Algorithmica*, 78(3):945–967, 2017. DOI: 10.1007/S00453-016-0186-Z.

[35] Pavel Klavík, Jan Kratochvíl, and Tomás Vyskočil. Extending partial representations of interval graphs. In Mitsunori Ogihara and Jun Tarui, editors, *Proc. 8th Annual Conference on Theory and Applications of Models of Computation (TAMC)*, volume 6648 of *LNCS*, pages 276–285. Springer, 2011. DOI: 10.1007/978-3-642-20877-5_28.

[36] Yasuaki Kobayashi, Yuto Okada, and Alexander Wolff. Recognizing 2-layer and outer k-planar graphs. In Oswin Aichholzer and Haitao Wang, editors, *Proc. 41st Annu. Sympos. Comput. Geom. (SoCG)*, volume 332 of *LIPIcs*, 65:1–65:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. DOI: 10.4230/LIPIcs.SoCG.2025.65.

[37] Tamás Kővari, Vera T. Sós, and Pál Turán. On a problem of K. Zarankiewicz. *Coll. Math.*, 3(1):50–57, 1954. URL: http://eudml.org/doc/210011.

[38] Krzysztof Koźmiński and Edwin Kinnen. Rectangular duals of planar graphs. *Networks*, 15(2):145–157, 1985. DOI: 10.1002/net.3230150202.

[39] Peter McMullen, Christoph Schulz, and Jörg M. Wills. Polyhedral 2-manifolds in E^3 with unusually large genus. *Israel J. Math.*, 46:127–144, 1983. DOI: 10.1007/BF02760627.

[40] Miriam Münch. *Partial and Simultaneous Representations of Circular Permutation Graphs*. Master's thesis, University of Passau, 2021.

[41] Christophe Paul and Ignaz Rutter. Circle graphs can be recognized in linear time. In Meena Mahajan, Florin Manea, Annabelle McIver, and Kim Thang Nguyen, editors, *Proc. 43rd International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 364 of *LIPIcs*, 72:1–72:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. DOI: 10.4230/LIPICS.STACS.2026.72.

[42] Bert Randerath, Ewald Speckenmeyer, Endre Boros, Peter L. Hammer, Alexander Kogan, Kazuhisa Makino, Bruno Simeone, and Ondrej Cepek. A satisfiability formulation of problems on level graphs. *Electron. Notes Discret. Math.*, 9:269–277, 2001. DOI: 10.1016/S1571-0653(04)00327-0.

[43] Fred S. Roberts. *Representations of Indifference Relations*. PhD thesis, Department of Mathematics, Stanford University, 1968.

[44] Irena Rusu. On the complexity of recognizing stick, biphook and max point-tolerance graphs. *Theor. Comput. Sci.*, 952:113773, 2023.

[45] Ajaykrishnan E. S., Robert Ganian, Daniel Lokshtanov, and Vaishali Surianarayanan. A quasi-polynomial time algorithm for 3-coloring circle graphs. In Sepehr Assadi and Eva Rotenberg, editors, *Proc. 9th Symposium on Simplicity in Algorithms (SOSA)*, pages 65–80. SIAM, 2026. DOI: 10.1137/1.9781611978964.6.

[46] Kazuhiro Sado and Yoshihide Igarashi. A function for evaluating the computing time of a bubbling system. *Theor. Comput. Sci.*, 54:315–324, 1987. DOI: 10.1016/0304-3975(87)90136-8.

[47] André Schulz. Side-contact representations with convex polygons in 3d: new results for complete bipartite graphs. In Michael A. Bekos and Markus Chimani, editors, *Proc. Graph Drawing and Network Visualization (GD)*, volume 14465 of *LNCS*, pages 296–303. Springer, 2023. DOI: 10.1007/978-3-031-49272-3_20.

[48] Peter Stumpf. *Partial Representation Extension and Simultaneous Representation of Intersection Graphs*. PhD thesis, Universität Passau, 2024.

[49] Walter Unger. The complexity of colouring circle graphs. In Alain Finkel and Matthias Jantzen, editors, *Proc. 9th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 577 of *LNCS*, pages 389–400. Springer, 1992. DOI: 10.1007/3-540-55210-3_199.

[50] Katsuhisa Yamanaka, Takashi Horiyama, Takeaki Uno, and Kunihiro Wasa. Ladder-lottery realization. In *Proc. 30th Canad. Conf. Comput. Geom. (CCCG)*, pages 61–67, 2018. URL: http://www.cs.umanitoba.ca/~cccg2018/papers/session2A-p3.pdf.

[51] Johannes Zink, Julian Walter, Joachim Baumeister, and Alexander Wolff. Layered drawing of undirected graphs with generalized port constraints. *Comput. Geom.*, 105–106(101886):1–29, 2022. DOI: 10.1016/j.comgeo.2022.101886.

4 Published Project Results

4.1 Category A – Articles in peer-reviewed journals, contributions to peer-reviewed conferences or to anthology volumes, and book publications

The majority of our publications have also appeared in a journal. Whenever possible, we have opted for venues that admit open access publications. In particular, all journal articles with the exception of [J5, 11, J21, J1, J19] are open access.

Journal publications

[J1] Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Ignaz Rutter. Beyond level planarity: cyclic, torus, and simultaneous level planarity. *Theor. Comput. Sci.*, 804:161–170, 2020. DOI: 10.1016/J.TCS.2019.11.024.

[J2] Patrizio Angelini, Ignaz Rutter, and T. P. Sandhya. Extending partial orthogonal drawings. *J. Graph Algorithms Appl.*, 25(1):581–602, 2021. DOI: 10.7155/JGAA.00573.

[J3] Elena Arseneva, Linda Kleist, Boris Klemz, Maarten Löffler, André Schulz, Birgit Vogtenhuber, and Alexander Wolff. Adjacency graphs of polyhedral surfaces. *Discrete Comput. Geom.*, 71:1429–1455, 2024. DOI: 10.1007/s00454-023-00537-6.

[J4] Patricia Bachmann, Ignaz Rutter, and Peter Stumpf. On 3-coloring circle graphs. *J. Graph Algorithms Appl.*, 28(1):389–402, 2024. DOI: 10.7155/JGAA.V28I1.2991.

[J5] Martin Balko, Steven Chaplick, Robert Ganian, Siddharth Gupta, Michael Hoffmann, Pavel Valtr, and Alexander Wolff. Bounding and computing obstacle numbers of graphs. *SIAM J. Discrete Math.*, 38(2):1537–1565, 2024. DOI: 10.1137/23M1585088.

[J6] Michael A. Bekos, Carla Binucci, Giuseppe Di Battista, Walter Didimo, Martin Gronemann, Karsten Klein, Maurizio Patrignani, and Ignaz Rutter. On turn-regular orthogonal representations. *J. Graph Algorithms Appl.*, 26(3):285–306, 2022. DOI: 10.7155/JGAA.00595.

[J7] Sujoy Bhore, Guangping Li, Martin Nöllenburg, Ignaz Rutter, and Hsiang-Yun Wu. Untangling circular drawings: algorithms and complexity. *Comput. Geom.*, 111:101975, 2023. DOI: 10.1016/J.COMGEO.2022.101975.

[J8] Thomas Bläsius, Simon D. Fink, and Ignaz Rutter. Synchronized planarity with applications to constrained planarity problems. *ACM Trans. Algorithms*, 19(4):34:1–34:23, 2023. DOI: 10.1145/3607474.

[J9] Guido Brückner and Ignaz Rutter. An spqr-tree-like embedding representation for level planarity. *J. Comput. Geom.*, 14(1):48–77, 2023. DOI: 10.20382/JOCG.V14I1A3.

[J10] Guido Brückner and Ignaz Rutter. Radial level planarity with fixed embedding. *J. Graph Algorithms Appl.*, 25(1):353–366, 2021. DOI: 10.7155/JGAA.00561.

[J11] Guido Brückner, Ignaz Rutter, and Peter Stumpf. Extending partial representations of circle graphs in near-linear time. *Algorithmica*, 86(7):2152–2173, 2024. DOI: 10.1007/S00453-024-01216-5.

[J12] Guido Brückner, Ignaz Rutter, and Peter Stumpf. Level-planarity: transitivity vs. even crossings. *Electron. J. Comb.*, 29(4), 2022. DOI: 10.37236/10814.

[J13] Steven Chaplick, Stefan Felsner, Philipp Kindermann, Jonathan Klawitter, Ignaz Rutter, and Alexander Wolff. Simple algorithms for partial and simultaneous rectangular duals with given contact orientations. *Theor. Comput. Sci.*, 919:66–74, 2022. DOI: 10.1016/J.TCS.2022.03.031.

[J14] Steven Chaplick, Philipp Kindermann, Andre Löffler, Florian Thiele, Alexander Wolff, Alexander Zaft, and Johannes Zink. Stick graphs with and without length constraints. *J. Graph Alg. Appl.*, 24(4):657–681, 2020. DOI: 10.7155/jgaa.00524.

[J15] Simon D. Fink, Matthias Pfretzschner, and Ignaz Rutter. Experimental comparison of pc-trees and pq-trees. *ACM J. Exp. Algorithmics*, 28:1.10:1–1.10:24, 2023. DOI: 10.1145/3611653.

[J16] Simon Dominik Fink and Ignaz Rutter. Constrained planarity in practice – engineering the synchronized planarity algorithm. *J. Graph Algorithms Appl.*, 29(1):91–123, 2025. DOI: 10.7155/JGAA.V29I1.2923.

[J17] Simon Dominik Fink and Ignaz Rutter. Maintaining triconnected components under node expansion. *Comput. Geom. Topol.*, 3(2):7:1–7:20, 2024. DOI: 10.57717/CGT.V3I2.44.

[J18] Oksana Firman, Philipp Kindermann, Boris Klemz, Alexander Ravsky, Alexander Wolff, and Johannes Zink. Deciding the feasibility and minimizing the height of tangles. *J. Graph Alg. Appl.*, 29(1):135–158, 2025. DOI: 10.7155/jgaa.v29i1.3053.

[J19] Emilio Di Giacomo, Giuseppe Liotta, Maurizio Patrignani, Ignaz Rutter, and Alessandra Tappini. Nodetrix planarity testing with small clusters. *Algorithmica*, 81(9):3464–3493, 2019. DOI: 10.1007/S00453-019-00585-6.

[J20] Giuseppe Liotta, Ignaz Rutter, and Alessandra Tappini. Parameterized complexity of graph planarity with restricted cyclic orders. *J. Comput. Syst. Sci.*, 135:125–144, 2023. DOI: 10.1016/J.JCSS.2023.02.007.

[J21] Giuseppe Liotta, Ignaz Rutter, and Alessandra Tappini. Simultaneous fpq-ordering and hybrid planarity testing. *Theor. Comput. Sci.*, 874:59–79, 2021. DOI: 10.1016/J.TCS.2021.05.012.

[J22] Giordano Da Lozzo and Ignaz Rutter. Planarity of streamed graphs. *Theor. Comput. Sci.*, 799:1–21, 2019. DOI: 10.1016/J.TCS.2019.09.029.

[J23] Miriam Münch, Ignaz Rutter, and Peter Stumpf. Partial and simultaneous transitive orientations via modular decompositions. *Algorithmica*, 86(4):1263–1292, 2024. DOI: 10.1007/S00453-023-01188-Y.

[J24] Tim Ophelders, Ignaz Rutter, Bettina Speckmann, and Kevin Verbeek. Polygon-universal graphs. *J. Comput. Geom.*, 16(1):488–516, 2025. DOI: 10.20382/JOCG.V16I1A14.

[J25] Marcel Radermacher, Klara Reichard, Ignaz Rutter, and Dorothea Wagner. Geometric heuristics for rectilinear crossing minimization. *ACM J. Exp. Algorithmics*, 24(1):1.12:1–1.12:21, 2019. DOI: 10.1145/3325861.

[J26] Marcel Radermacher and Ignaz Rutter. Inserting an edge into a geometric embedding. *Comput. Geom.*, 102:101843, 2022. DOI: 10.1016/J.COMGEO.2021.101843.

[J27] Ignaz Rutter, Darren Strash, Peter Stumpf, and Michael Vollmer. Simultaneous representation of proper and unit interval graphs. *Algorithmica*, 87(5):783–811, 2025. DOI: 10.1007/S00453-025-01296-X.

Conference publications

[C1] Elena Arseneva, Linda Kleist, Boris Klemz, Maarten Löffler, André Schulz, Birgit Vogtenhuber, and Alexander Wolff. Adjacency graphs of polyhedral surfaces. In Kevin Buchin and Éric Colin de Verdière, editors, *Proc. 37th Annu. Sympos. Comput. Geom. (SoCG)*, volume 189 of *LIPIcs*, 11:1–17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. DOI: 10.4230/LIPIcs.SoCG.2021.11.

[C2] Patricia Bachmann, Ignaz Rutter, and Peter Stumpf. On 3-coloring circle graphs. In Michael A. Bekos and Markus Chimani, editors, *Graph Drawing and Network Visualization - 31st International Symposium (GD)*, volume 14465 of *LNCS*, pages 152–160. Springer, 2023. DOI: 10.1007/978-3-031-49272-3_11.

[C3] Martin Balko, Steven Chaplick, Robert Ganian, Siddharth Gupta, Michael Hoffmann, Pavel Valtr, and Alexander Wolff. Bounding and computing obstacle numbers of graphs. In Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman, editors, *Proc. 30th Europ. Symp. Algorithms (ESA)*, volume 244 of *LIPIcs*, 11:1–13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. DOI: 10.4230/LIPIcs.ESA.2022.11.

[C4] Michael A. Bekos, Carla Binucci, Giuseppe Di Battista, Walter Didimo, Martin Gronemann, Karsten Klein, Maurizio Patrignani, and Ignaz Rutter. On turn-regular orthogonal representations. In David Auber and Pavel Valtr, editors, *Proc. 28th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 12590 of *LNCS*, pages 250–264. Springer, 2020. DOI: 10.1007/978-3-030-68766-3_20.

[C5] Sujoy Bhore, Guangping Li, Martin Nöllenburg, Ignaz Rutter, and Hsiang-Yun Wu. Untangling circular drawings: algorithms and complexity. In Hee-Kap Ahn and Kunihiko Sadakane, editors, *Proc. 32nd International Symposium on Algorithms and Computation (ISAAC)*, volume 212 of *LIPIcs*, 19:1–19:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. DOI: 10.4230/LIPICS.ISAAC.2021.19.

[C6] Thomas Bläsius, Simon D. Fink, and Ignaz Rutter. Synchronized planarity with applications to constrained planarity problems. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *Proc. 29th Annual European Symposium on Algorithms (ESA)*, volume 204 of *LIPIcs*, 19:1–19:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. DOI: 10.4230/LIPICS.ESA.2021.19.

[C7] Guido Brückner, Markus Himmel, and Ignaz Rutter. An spqr-tree-like embedding representation for upward planarity. In Daniel Archambault and Csaba D. Tóth, editors, *Proc. 27th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 11904 of *LNCS*, pages 517–531. Springer, 2019. DOI: 10.1007/978-3-030-35802-0_39.

[C8] Guido Brückner and Ignaz Rutter. An spqr-tree-like embedding representation for level planarity. In Yixin Cao, Siu-Wing Cheng, and Minming Li, editors, *Proc. 31st International Symposium on Algorithms and Computation (ISAAC)*, volume 181 of *LIPIcs*, 8:1–8:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. DOI: 10.4230/LIPICS.ISAAC.2020.8.

[C9] Guido Brückner, Ignaz Rutter, and Peter Stumpf. Extending partial representations of circle graphs in near-linear time. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *Proc. 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 241 of *LIPIcs*, 25:1–25:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. DOI: 10.4230/LIPICS.MFCS.2022.25.

[C10] Steven Chaplick, Philipp Kindermann, Jonathan Klawitter, Ignaz Rutter, and Alexander Wolff. Extending partial representations of rectangular duals with given contact orientations. In Tiziana Calamoneri and Federico Coró, editors, *Proc. 12th International Conference on Algorithms and Complexity (CIAC)*, volume 12701 of *LNCS*, pages 340–353. Springer, 2021. DOI: 10.1007/978-3-030-75242-2_24.

[C11] William Evans, Paweł Rzażewski, Noushin Saeedi, Chan-Su Shin, and Alexander Wolff. Representing graphs and hypergraphs by touching polygons in 3D. In Daniel Archambault and Csaba D. Tóth, editors, *Proc. 27th Int. Symp. Graph Drawing & Network Vis. (GD)*, volume 11904 of *LNCS*, pages 18–32. Springer, 2019. DOI: 10.1007/978-3-030-35802-0_2.

[C12] Jiří Fiala, Oksana Firman, Giuseppe Liotta, Alexander Wolff, and Johannes Zink. Outerplanar and forest storyplans. In Henning Fernau, Serge Gaspers, and Ralf Klasing, editors, *Proc. 50th Int. Conf. Current Trends Theory & Practice Comput. Sci. (SOFSEM)*, volume 14519 of *LNCS*, pages 211–225. Springer, 2024. DOI: 10.1007/978-3-031-52113-3_15.

[C13] Jirí Fiala, Ignaz Rutter, Peter Stumpf, and Peter Zeman. Extending partial representations of circular-arc graphs. In Michael A. Bekos and Michael Kaufmann, editors, *Proc. 48th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 13453 of *LNCS*, pages 230–243. Springer, 2022. DOI: 10.1007/978-3-031-15914-5_17.

[C14] Simon D. Fink, Matthias Pfretzschner, and Ignaz Rutter. Experimental comparison of pc-trees and pq-trees. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *Proc. 29th Annual European Symposium on Algorithms (ESA)*, volume 204 of *LIPIcs*, 43:1–43:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. DOI: 10.4230/LIPICS.ESA.2021.43.

[C15] Simon D. Fink, Matthias Pfretzschner, and Ignaz Rutter. Parameterized complexity of simultaneous planarity. In Michael A. Bekos and Markus Chimani, editors, *Graph Drawing and Network Visualization - 31st International Symposium (GD)*, volume 14466 of *LNCS*, pages 82–96. Springer, 2023. DOI: 10.1007/978-3-031-49275-4_6.

[C16] Simon D. Fink and Ignaz Rutter. Constrained planarity in practice: engineering the synchronized planarity algorithm. In Rezaul Chowdhury and Solon P. Pissis, editors, *Proc. Symposium on Algorithm Engineering and Experiments (ALENEX)*, pages 1–14. SIAM, 2024. DOI: 10.1137/1.9781611977929.1.

[C17] Simon D. Fink and Ignaz Rutter. Maintaining triconnected components under node expansion. In Marios Mavronicolas, editor, *Algorithms and Complexity - 13th International Conference (CIAC)*, volume 13898 of *LNCS*, pages 202–216. Springer, 2023. DOI: 10.1007/978-3-031-30448-4_15.

[C18] Simon D. Fink, Ignaz Rutter, and Sandhya T. P. A simple partially embedded planarity test based on vertex-addition. In Ioana Oriana Bercea and Rasmus Pagh, editors, *Proc. Symposium on Simplicity in Algorithms (SOSA)*, pages 496–508. SIAM, 2025. DOI: 10.1137/1.9781611978315.37.

[C19] Oksana Firman, Philipp Kindermann, Boris Klemz, Alexander Ravsky, Alexander Wolff, and Johannes Zink. The complexity of finding tangles. In Leszek Gasieniec, editor, *Proc. 49th Int. Conf. Current Trends Theory & Practice Comput. Sci. (SOFSEM)*, volume 13878 of *LNCS*, pages 3–17. Springer, 2023. DOI: 10.1007/978-3-031-23101-8_1.

[C20] Grzegorz Gutowski, Konstanty Junosza-Szaniawski, Felix Klesen, Paweł Rzażewski, Alexander Wolff, and Johannes Zink. Coloring and recognizing mixed interval graphs. In Satoru Iwata and Naonori Kakimura, editors, *Proc. 34th Annu. Int. Symp. Algorithms Comput. (ISAAC)*, volume 283 of *LIPIcs*, 36:1–14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. DOI: 10.4230/LIPIcs.ISAAC.2023.36.

[C21] Grzegorz Gutowski, Florian Mittelstädt, Ignaz Rutter, Joachim Spoerhase, Alexander Wolff, and Johannes Zink. Coloring mixed and directional interval graphs. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Proc. 30th International Symposium on Graph Drawing and Network Visualization (GD)*, LNCS, pages 418–431. Springer, 2022. DOI: 10.1007/978-3-031-22203-0_30.

[C22] Giuseppe Liotta, Ignaz Rutter, and Alessandra Tappini. Parameterized complexity of graph planarity with restricted cyclic orders. In Michael A. Bekos and Michael Kaufmann, editors, *Proc. 48th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 13453 of *LNCS*, pages 383–397. Springer, 2022. DOI: 10.1007/978-3-031-15914-5_28.

[C23] Giuseppe Liotta, Ignaz Rutter, and Alessandra Tappini. Simultaneous fpq-ordering and hybrid planarity testing. In Alexander Chatzigeorgiou, Riccardo Dondi, Herodotos Herodotou, Christos A. Kapoutsis, Yannis Manolopoulos, George A. Papadopoulos, and Florian Sikora, editors, *Proc. 46th International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM)*, volume 12011 of *LNCS*, pages 617–626. Springer, 2020. DOI: 10.1007/978-3-030-38919-2_51.

[C24] Miriam Münch, Ignaz Rutter, and Peter Stumpf. Partial and simultaneous transitive orientations via modular decompositions. In Sang Won Bae and Heejin Park, editors, *Proc. 33rd International Symposium on Algorithms and Computation (ISAAC)*, volume 248 of *LIPIcs*, 51:1–51:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. DOI: 10.4230/LIPICS.ISAAC.2022.51.

[C25] Tim Ophelders, Ignaz Rutter, Bettina Speckmann, and Kevin Verbeek. Polygon-universal graphs. In Kevin Buchin and Éric Colin de Verdière, editors, *Proc. 37th International Symposium on Computational Geometry (SoCG)*, volume 189 of *LIPIcs*, 55:1–55:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. DOI: 10.4230/LIPICS.SOCG.2021.55.

[C26] Marcel Radermacher and Ignaz Rutter. Geometric crossing-minimization - A scalable randomized approach. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *Proc. 27th Annual European Symposium on Algorithms (ESA)*, volume 144 of *LIPIcs*, 76:1–76:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. DOI: 10.4230/LIPICS.ESA.2019.76.

[C27] Marcel Radermacher, Ignaz Rutter, and Peter Stumpf. Towards a characterization of stretchable aligned graphs. In David Auber and Pavel Valtr, editors, *Proc. 28th International Symposium on Graph Drawing and Network Visualization (GD)*, volume 12590 of *LNCS*, pages 295–307. Springer, 2020. DOI: 10.1007/978-3-030-68766-3_23.

[C28] Ignaz Rutter, Darren Strash, Peter Stumpf, and Michael Vollmer. Simultaneous representation of proper and unit interval graphs. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *Proc. 27th Annual European Symposium on Algorithms (ESA)*, volume 144 of *LIPIcs*, 80:1–80:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. DOI: 10.4230/LIPICS.ESA.2019.80.

[C29] Ignaz Rutter and Peter Stumpf. Simultaneous representation of interval graphs in the sunflower case. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *Proc. 31st Annual European Symposium on Algorithms (ESA)*, volume 274 of *LIPIcs*, 90:1–90:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. DOI: 10.4230/LIPICS.ESA.2023.90.

4.2 Category B – Any other form of published results

Informal workshop publications and poster abstracts

[I1] Simon D. Fink, Matthias Pfretzschner, Ignaz Rutter, and Marie D. Sieper. Clustered planarity variants for level graphs. In Michael A. Bekos and Charis Papadopoulos, editors, *Booklet of Abstracts of the 40th European Workshop on Computational Geometry (EuroCG)*, 52:1–9, 2024.

[I2] Simon D. Fink, Matthias Pfretzschner, Ignaz Rutter, and Marie Diana Sieper. Clustered planarity variants for level graphs. *CoRR*, abs/2402.13153, 2024. arXiv: 2402.13153.