Report from Dagstuhl Seminar 23162

New Frontiers of Parameterized Complexity in Graph Drawing

Robert Ganian[*1], **Fabrizio Montecchiani**[*2], **Martin Nöllenburg**[*3], **Meirav Zehavi**[*4], **and Liana Khazaliya**[†5]

1 **TU Wien, AT.** rganian@gmail.com
2 **University of Perugia, IT.** fabrizio.montecchiani@unipg.it
3 **TU Wien, AT.** noellenburg@ac.tuwien.ac.at
4 **Ben Gurion University – Beer Sheva, IL.** zehavimeirav@gmail.com
5 **TU Wien, AT.** liana.khazaliya@gmail.com

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 23162 "New Frontiers of Parameterized Complexity in Graph Drawing". The seminar was held in-person from April 16 to April 21, 2023. It brought together 32 researchers from the Graph Drawing and the Parameterized Complexity research communities to discuss and explore new research frontiers on the interface between the two fields. The report collects the abstracts of talks and open problems presented in the seminar, as well as brief progress reports from the working groups.

Seminar April 16–21, 2023 – https://www.dagstuhl.de/23162
2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Computational geometry
Keywords and phrases algorithm design, computational geometry, graph drawing, parameterized complexity
Digital Object Identifier 10.4230/DagRep.13.4.58

1 Executive Summary

Robert Ganian (TU Wien, AT)
Fabrizio Montecchiani (University of Perugia, IT)
Martin Nöllenburg (TU Wien, AT)
Meirav Zehavi (Ben Gurion University – Beer Sheva, IL)

In modern life, it is of paramount importance that computational tasks be performed both precisely and efficiently. Thus, the design and analysis of algorithms to execute such tasks lie at the heart of computer science. However, since the proof of the Cook-Levin theorem in the 1970s, numerous problems have been shown to be NP-hard. Fortunately, the field of *parameterized complexity* [7, 5, 10, 20], initiated in the 1980s by Downey and Fellows, yields (in)tractability results that are both deeper and fine-grained. Specifically, it shows that the "hardness" of an NP-hard problem can often be traced to particular *parameters* of its instances.

[*] Editor / Organizer
[†] Editorial Assistant / Collector

New Frontiers of Parameterized Complexity in Graph Drawing, *Dagstuhl Reports*, Vol. 13, Issue 4, pp. 58–97
Editors: Robert Ganian, Fabrizio Montecchiani, Martin Nöllenburg, Meirav Zehavi, and Liana Khazaliya
DAGSTUHL REPORTS Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In a nutshell, a *parameterization* of a problem Π is the association of a parameter k with every instance of Π, capturing how "structured" the instance is. The main goal in parameterized complexity is to confine the combinatorial explosion in the runtime to a suitable parameter k, rather than to let it depend on the entire input size n. So, fixed-parameter algorithms naturally "scale" with the amount of structure of a given instance. Formally, a problem is *fixed-parameter tractable* (in short, in FPT) with respect to a parameter k if it can be solved by an algorithm, called a *fixed-parameter algorithm*, whose runtime is bounded by $f(k) \cdot n^{O(1)}$ for some computational function f of k. Over the past four decades, the parameterized complexity paradigm has yielded a rich and deep theory, with powerful approaches to obtain fixed-parameter algorithms for a variety of problems as well as machinery that can rule out such algorithms. For problems in FPT, it also offers the necessary tools to develop the fastest possible fixed-parameter algorithms (often with tight conditional lower bounds).

While the parameterized paradigm can be applied in a wide range of different fields, the focus of this Dagstuhl Seminar lay squarely on its potential synergies with *graph drawing*, a self-standing discipline that has evolved tremendously over the last decades. Indeed, given the ubiquity of graphs in many fields of science and technology, there is a strong interest in algorithms that can provide effective graphical representations of graphs, for the sake of both analysis and communication. Today, graph drawing is a mature area of computer science [6, 17, 21, 22] with its own annual conference, the International Symposium on Graph Drawing and Network Visualization (GD)[1]. The focus of graph drawing as a research area today is on both fundamental and practical aspects, such as combinatorics and algorithm design on the one hand, and the development of network visualization systems and interfaces on the other hand. At a very high level, graph drawing deals with the construction and analysis of geometric representations of graphs and networks subject to specific layout conventions and constraints, such as different notions of planarity or more general crossing constraints, grid layouts, orthogonal drawings, and many more. Notably, this gives rise to many computational problems that are NP-hard in the classical sense but naturally multivariate, making parameterized analysis particularly attractive. Yet, so far, research at the intersection of parameterized complexity and graph drawing has not reached its full potential.

Parameterized Graph Drawing. Most of the early efforts to conduct research at the intersection of parameterized complexity and graph drawing have been directed at variants of the classic Crossing Minimization problem, introduced by Turán in 1940 [23], parameterized by the number of crossings. Here, the objective is to draw a given graph in the plane so as to induce the minimum number of crossings, which was shown to be FPT already in 2001 [13]. A few subsequent works followed [18, 15], including the best paper of GD 2019 [16]. Other examples of graph drawing problems successfully targeted by the parameterized paradigm include crossing minimization in restricted settings [8, 19], crossing-sensitive subgraph detection [1, 14], parameterized algorithms for stack and queue layouts [2, 3, 4], and drawing extension problems [9, 11]. While these and other works already lay down the foundations of possible positive synergies between the two fields, there still is a multitude of problems and opportunities in graph drawing where parameterized analysis would be beneficial and has not yet been carried out.

[1] see www.graphdrawing.org

Seminar Goals

The main goal of the seminar was to explore and initiate collaborations at the intersection of Graph Drawing and Parameterized Complexity, with emphasis on new research frontiers. In particular:

- First, the seminar focused on prominent topics in Graph Drawing, encouraging the consideration of topics that have been only little – or not at all – studied from the perspective of Parameterized Complexity. Here, the discussions centered on the identification of open problems of wide interest as well as directions for future research.
- Second, the seminar focused on new tools and sub-areas of Parameterized Complexity, such as structural parameterization, non-standard general decomposition theorems, and FPT-approximation, which have been rarely – or not at all – examined in the context of Graph Drawing. The aim was to understand their potential and relevance in the context of the aforementioned topics.

The seminar included invited talks focused on the above goals, setting the ground for the participants to propose problems (and, more generally, research directions) to work on, as well as be familiar with relevant tools to work with. The final selection of problems targeted by working groups was carried out during the seminar itself. This, in turn, also resulted in the establishment of new and fruitful collaborations.

Seminar Program

We started the seminar week on Monday morning with short self-introduction presentations of all participants, followed by the four invited overview talks mentioned above, which aimed to set the grounds for the research discussions throughout the week. We invited two speakers from each of the two communities coming together in this seminar to create a common understanding of research questions, tools, and terminology. The first of these lectures was given by Saket Saurabh on the topic of fixed-parameter tractability for geometric intersection graphs. Next, Ignaz Rutter spoke about the widely used SPQR tree data structure for representing planar graph embeddings. As the third speaker, Bart M. P. Jansen highlighted three frontiers in parameterized graph drawing based on lossy (structural) kernelization and hybrid parameterizations. Lastly, Giordano Da Lozzo presented complexity results and parameterized algorithms for the upward book embedding problem. Further, we had a short session for reports about the final results of working groups from the previous edition of the seminar in 2021. Section 3 contains more detailed abstracts of the invited lectures and the reports from Dagstuhl Seminar 21293.

The remainder of the seminar focused on research discussions on new frontiers of parameterized graph drawing. In order to identify these frontiers, we had two open problem sessions, one on Monday afternoon and one on Tuesday morning. A total of 13 open research questions (see Section 4) were contributed by the participants. Based on a preference vote, we created five working groups for the seminar week, each having six or seven participants mixed from the two communities joined in this seminar. The following five topics were investigated in depth:

1. Parameterized Complexity of the Maximum Bimodal Subgraph Problem
2. Parameterized Complexity of Upward Planarity and st-Planar Edge Completion Problems
3. Parameterized Algorithms for the Sequential Crossing Number Problem
4. Deleting Few Edges from Ordered Graphs to Make them k-Plane
5. Clustered Planarity and Planar $\mathcal{F}$-Augmentations

For each of the five working groups, Section 5 contains a detailed report of their current state of research. During the seminar, on Tuesday and Thursday afternoon, we had plenary sessions for the groups' progress reports sharing their latest results and further plans. Wednesday afternoon was reserved for the traditional excursion, for which we enjoyed a Treetop-Walk near the Great Bend in the Saar and then hiked to Mettlach for a traditional brewery dinner.

Future Plans

The composition of the working groups was made having in mind two main criteria: balancing the number of participants from the two communities, guaranteeing the presence of some young researcher in each group. This strategy was indeed successful and it led to new collaborations between researchers in the graph drawing and parameterized complexity communities, as well as to new opportunities for young researchers. As a primary outcome of the seminar, we expect research papers published at the core conferences and journals for the graph drawing and parameterized complexity communities, for instance:

- The International Symposium on Computational Geometry (SoCG),
- The International Symposium on Graph Drawing and Network Visualization (GD),
- The ACM-SIAM Symposium on Discrete Algorithms (SODA), and
- The International Symposium on Algorithms and Computation (ISAAC).

In the mid- and long-term horizon, the seminar will also help to gradually blend the boundary between the two communities, with young researchers building their academic journey on the intersection of the two fields. In this respect, the seminar can also lead to the development of new parameterized tools and techniques that are designed to deal with the specific obstacles that arise when trying to apply parameterized approaches in the graph drawing setting.

This Dagstuhl Seminar and its predecessor (Dagstuhl Seminar 21293) have revealed, in a systematic way, the astounding wealth of problems in Graph Drawing that are naturally multivariate and hence suitable for parameterized analysis. The two communities are now aware of these problems, and the above mentioned conferences (among others) publish every year a growing number of results making progress on these problems. Besides this, informal feedback from the communities confirms the great interest in having such seminars on a more regular basis. We therefore plan to have follow-up seminars to support the growth of parameterized complexity in graph drawing.

Evaluation

Taking into account the results of the Dagstuhl survey conducted after the seminar and informal feedback to the organizers, it is fair to say that the seminar was highly appreciated. The participants provided significantly above-average scores for the seminar, particularly in areas concerning new collaborative research projects, identification of novel research directions, and collaboration between different research communities. Overall, we believe that the seminar's goals of identifying new research directions and initiating collaborations at the intersection of the two different fields of Graph Drawing and Parameterized Complexity were very successful. We are looking forward to seeing the first scientific outcomes of the seminar in the near future and to continuing the efforts to support the growth of interest in parameterized analysis of problems in Graph Drawing.

Acknowledgments

Schloss Dagstuhl, once again, was a perfect location for hosting this research seminar with its great conference and meeting facilities, combined with the unique atmosphere of the castle and lots of opportunities for socializing and networking. On behalf of all participants, we want to express our gratitude to the entire Dagstuhl staff for letting us have such a wonderful week. We further thank Liana Khazaliya for helping us collect the various contributions and prepare this report.

References

1 Akanksha Agrawal, Grzegorz Guspiel, Jayakrishnan Madathil, Saket Saurabh, and Meirav Zehavi. Connecting the Dots (with Minimum Crossings). In Gill Barequet and Yusu Wang, editors, *Symposium on Computational Geometry (SoCG 2019)*, volume 129 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:17, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

2 Michael J. Bannister, David Eppstein, and Joseph A. Simons. Fixed parameter tractability of crossing minimization of almost-trees. In *Graph Drawing (GD 2013)*, volume 8242 of *Lecture Notes in Computer Science*, pages 340–351. Springer, 2013.

3 Sujoy Bhore, Robert Ganian, Fabrizio Montecchiani, and Martin Nöllenburg. Parameterized algorithms for book embedding problems. *J. Graph Algorithms Appl.*, 24(4):603–620, 2020.

4 Sujoy Bhore, Robert Ganian, Fabrizio Montecchiani, and Martin Nöllenburg. Parameterized algorithms for queue layouts. In *Graph Drawing*, volume 12590 of *Lecture Notes in Computer Science*, pages 40–54. Springer, 2020.

5 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

6 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice-Hall, 1999.

7 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity.* Texts in Computer Science. Springer Verlag, 2013.

8 Vida Dujmovic, Michael R. Fellows, Matthew Kitching, Giuseppe Liotta, Catherine McCartin, Naomi Nishimura, Prabhakar Ragde, Frances A. Rosamond, Sue Whitesides, and David R. Wood. On the parameterized complexity of layered graph drawing. *Algorithmica*, 52(2):267–292, 2008.

9 Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, and Martin Nöllenburg. Extending partial 1-planar drawings. In *ICALP*, volume 168 of *LIPIcs*, pages 43:1–43:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

10 F. V. Fomin, D. Lokshtanov, S. Saurabh, and M. Zehavi. *Kernelization: Theory of Parameterized Preprocessing.* Cambridge University Press, 2018.

11 Robert Ganian, Thekla Hamm, Fabian Klute, Irene Parada, and Birgit Vogtenhuber. Crossing-optimal extension of simple drawings. In *ICALP*, volume 198 of *LIPIcs*, pages 72:1–72:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

12 Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.

13 Martin Grohe. Computing crossing numbers in quadratic time. *J. Comput. Syst. Sci.*, 68(2):285–302, 2004.

14 Magnús M. Halldórsson, Christian Knauer, Andreas Spillner, and Takeshi Tokuyama. Fixed-parameter tractability for non-crossing spanning trees. In *Algorithms and Data Structures (WADS 2007)*, volume 4619 of *Lecture Notes in Computer Science*, pages 410–421. Springer, 2007.

15 Petr Hlinený and Marek Dernár. Crossing number is hard for kernelization. In *Symposium on Computational Geometry (SoCG 2016)*, volume 51 of *LIPIcs*, pages 42:1–42:10. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2016.

16 Petr Hlinený and Abhisekh Sankaran. Exact crossing number parameterized by vertex cover. In *Graph Drawing and Network Visualization (GD 2019)*, volume 11904 of *Lecture Notes in Computer Science*, pages 307–319. Springer, 2019.

17 Michael Jünger and Petra Mutzel, editors. *Graph Drawing Software*. Springer, 2004.

18 Ken-ichi Kawarabayashi and Bruce A. Reed. Computing crossing number in linear time. In *Symposium on Theory of Computing (STOC 2007)*, pages 382–390. ACM, 2007.

19 Fabian Klute and Martin Nöllenburg. Minimizing crossings in constrained two-sided circular graph layouts. *J. Comput. Geom.*, 10(2):45–69, 2019.

20 Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics and Its Applications. OUP Oxford, 2006.

21 Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, 2004.

22 Roberto Tamassia, editor. *Handbook on Graph Drawing and Visualization*. Chapman and Hall/CRC, 2013.

23 Paul Turán. A note of welcome. *Journal of Graph Theory*, 1(1):7–9, 1977.

2 Table of Contents

Executive Summary

Overview of Talks

Open problems

Working groups

3 Overview of Talks

3.1 The Hitchhiker's Guide to SPQR-trees

Ignaz Rutter (Universität Passau, DE)

SPQR-trees are a widely used tool in Graph Drawing that is at the heart of a large body of algorithmic results that are concerned with embeddings of biconnected planar graphs. Their key property is that they break down the complicated task of choosing a planar embedding of a biconnected graph into multiple independent choices, each of which is either binary or consists in arbitrarily choosing a permutation of a set of parallel subgraphs between two pole vertices.

The talk gives an introduction to SPQR-trees that is aimed at a broad audience. To cater to different backgrounds, we first give three different definitions of SPQR-trees that are complementary to each other with respect to the properties that can be easily deduced from them. In the second part of the talk we illustrate how to use SPQR-trees to solve constrained embedding problems, i.e., how to find embeddings of planar graphs that satisfy certain constraints. The talk finishes with a brief discussion of extensions of SPQR-trees to planarity variants for directed graphs, such as upward- an level-planarity, where similar structures can be defined for single-source graphs.

3.2 New frontiers for graph drawing: Lossy (structural) kernelization and hybrid parameterizations

Bart M. P. Jansen (TU Eindhoven, NL)

URL https://www.win.tue.nl/ bjansen/talks/FrontiersGraphDrawing.pptx

The talk covers three frontiers for the study of graph drawing problems from the perspective of parameterized complexity. The frontiers are illustrated by examples for 2-Layer Planarization, a fruitfly for parameterized graph drawing. Additionally, a sample result on each frontier is given based on my recent work.

The first frontier consists of kernelization for structural parameters. What is the strongest parameterization that admits a polynomial kernel? This question was recently resolved for the undirected Feedback Vertex Set problem, using the notion of elimination distance.

The second frontier is lossy kernelization, which aims to reduce instances in such a way that an approximate solution to the reduced instance can be lifted to a (slightly worse) approximation to the original. A constant-factor lossy kernel of polynomial size was recently developed for Vertex Planarization.

The last frontier consists of hybrid parameterizations improving simultaneously on parameterizations by solution size and treewidth. Fixed-parameter tractable algorithms for such parameterizations were recently found for Vertex Planarization. The running time of this algorithm is not much worse than for solution-size parameterizations.

3.3 Upward Book Embeddings: Complexity and Parameterized Algorithms

Giordano Da Lozzo (University of Rome III, IT)

A *k-page upward book embedding* (kUBE) of a directed acyclic graph (a DAG, for short) is a book embedding of the DAG on k pages with the additional requirement that the vertices appear in a topological ordering along the spine of the book. The *page number* of a DAG is the minimum k for which it admits a kUBE. In 1999, Heath and Pemmaraju conjectured that the recognition problem of DAGs with page number 2, called 2UBE Testing, is NP-complete [3]. After 23 years, Bekos et al. have finally settled this conjecture in the affirmative [1]. In particular, this NP-completeness result holds even for planar st-graphs. On the algorithmic side, Binucci et al. have complemented the previous negative result by providing an $O(f(\beta) \cdot n + n^3)$-time algorithm for 2UBE Testing of planar st-graphs, where β is the branchwidth of the input graph and f is a singly-exponential function on β [2]. In this talk, we will review the previously mentioned main advancements in the complexity of 2UBE Testing. Moreover, we will survey interesting variants of the problem and highlight promising research directions in the study of parameterized algorithms for these variants.

References

1 Michael A. Bekos, Giordano Da Lozzo, Fabrizio Frati, Martin Gronemann, Tamara Mchedlidze, Chrysanthi N. Raftopoulou: Recognizing DAGs with page-number 2 is NP-complete. Theor. Comput. Sci. 946: 113689 (2023)

2 Carla Binucci, Giordano Da Lozzo, Emilio Di Giacomo, Walter Didimo, Tamara Mchedlidze, Maurizio Patrignani: Upward Book Embeddings of st-Graphs. SoCG 2019: 13:1-13:22

3 Lenwood S. Heath, Sriram V. Pemmaraju: Stack and Queue Layouts of Directed Acyclic Graphs: Part II. SIAM J. Comput. 28(5): 1588-1626 (1999)

3.4 Report from the Previous Edition: Product structure for h-framed graphs

Petr Hlinený (Masaryk University – Brno, CZ)

Joint work of Michael A. Bekos, Giordano Da Lozzo, Petr Hliněný, Michael Kaufmann
Main reference Michael A. Bekos, Giordano Da Lozzo, Petr Hlinený, Michael Kaufmann: "Graph Product Structure for h-Framed Graphs", in Proc. of the 33rd International Symposium on Algorithms and Computation, ISAAC 2022, December 19-21, 2022, Seoul, Korea, LIPIcs, Vol. 248, pp. 23:1–23:15, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.
URL https://doi.org//10.4230/LIPIcs.ISAAC.2022.23

The focus of the working group from the 2021 seminar was on the product structure theorem of Dujmovic, Joret, Micek, Morin, Ueckerdt, and Wood (J. ACM 2020). We were following two research directions, to find nontrivial direct algorithmic applications of the product structure theory (which do not seem to exist in this strict sense), and to suitably extend the scope of the product structure beyond planarity, while still giving explicit small bounds. We succeeded in the second direction as follows.

A graph is *h-framed*, if it admits a drawing in the plane whose uncrossed edges induce a biconnected spanning plane graph with faces of size at most h. For instance, every 1-planar graph is a subgraph of a 4-framed graph, and every h-framed graph is $O(h^2)$-planar. As the main result, we have proved:

▶ **Theorem.** *Let G be a not-necessarily simple h-framed graph with $h \geq 3$. Then the simplification of G is a subgraph of the strong product $H \boxtimes P \boxtimes K_q$, where H is a planar graph with $tw(H) \leq 3$, $q = 3\lfloor \frac{h}{2} \rfloor + \lfloor \frac{h}{3} \rfloor - 1$, and P is a path.*

We have used this theorem to improve known upper bounds on the queue number and the non-repetitive chromatic number on 1-planar and optimal 2-planar graphs, and to give explicit upper bounds on the twin-width of these graphs.

3.5 Report from the Previous Edition: Upward Planarity Testing

Kirill Simonov (Hasso-Plattner-Institut, Universität Potsdam, DE)

Joint work of Kirill Simonov, Michael A. Bekos, Giordano Da Lozzo, Petr Hliněný, Michael Kaufmann
Main reference Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Robert Ganian, Chrysanthi N. Raftopoulou, Kirill Simonov: "Parameterized Algorithms for Upward Planarity", CoRR, Vol. abs/2203.05364, 2022.
URL https://doi.org//10.48550/arXiv.2203.05364

In Upward Planarity Testing, the input is a directed acyclic graph (DAG) G and the question is whether there exists a planar drawing of G such that all edges are drawn upward, i.e., all edges monotonically increase in the vertical direction. Upward Planarity Testing has been extensively studied in the literature, and arises naturally in a number of situations where the aim is to obtain easy-to-parse planar representations of DAGs. The problem has been shown to be NP-complete already 25 years ago [5], and there are a few special cases known when the problem can be solved in polynomial time. Among others, when G is restricted to class of single-source DAGs [1], or the class of orientations of series-parallel graphs [4]. There have also been a couple of fixed-parameter tractability results for Upward Planarity Testing.

It was the goal of our working group at the previous edition of the seminar to push the boundaries Upward Planarity Testing from the parameterized complexity perspective, generalizing and unifying the previous results. Our first target was to generalize the polynomial-time single-source algorithm [1], which resulted in a single-exponential FPT algorithm when parameterized by the number of sources [2]. We also considered the problem parameterized by prominent structural parameters. We showed that Upward Planarity Testing is XP parameterized by treewidth, and FPT parameterized by treedepth of the underlying undirected graph [2]. All three algorithms are obtained using a novel framework for the problem that combines SPQR tree-decompositions with parameterized techniques. Finally, we used our improved analysis on the series-parallel case: We showed that Upward Planarity Testing can be solved in time $O(n^2)$ on n-vertex series-paralled graphs [3], while the previous-best algorithm runs in time $O(n^4)$ [4].

References

1 Paola Bertolazzi, Giuseppe Di Battista, Carlo Mannino, Roberto Tamassia: *Optimal Upward Planarity Testing of Single-Source Digraphs*. SIAM J. Comput. 27(1): 132-169 (1998)

2 Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Robert Ganian, Chrysanthi N. Raftopoulou, Kirill Simonov: *Parameterized Algorithms for Upward Planarity*. SoCG 2022: 26:1–26:16

3 Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Robert Ganian, Chrysanthi N. Raftopoulou, Kirill Simonov: *Testing Upward Planarity of Partial 2-Trees*. GD 2022: 175-187

4 Walter Didimo, Francesco Giordano, Giuseppe Liotta: *Upward Spirality and Upward Planarity Testing*. SIAM J. Discret. Math. 23(4): 1842-1899 (2009)

5 Ashim Garg, Roberto Tamassia: *On the computational complexity of upward and rectilinear planarity testing*. SIAM J. Comput. 31(2):601–625 (2001)

3.6 Report from the Previous Edition: A Parameterized Approach to Orthogonal Compaction

Siddharth Gupta (University of Warwick – Coventry, GB)

Joint work of Siddharth Gupta, Philipp Kindermann, Walter Didimo, Giuseppe Liotta, Alexander Wolff, Meirav Zehavi
Main reference Walter Didimo, Siddharth Gupta, Philipp Kindermann, Giuseppe Liotta, Alexander Wolff, Meirav Zehavi: "Parameterized Approaches to Orthogonal Compaction", CoRR, Vol. abs/2210.05019, 2022.
URL https://doi.org//10.48550/arXiv.2210.05019

Orthogonal graph drawings are used in applications such as UML diagrams, VLSI layout, cable plans, and metro maps. We focus on drawing planar graphs and assume that we are given an *orthogonal representation* that describes the desired shape, but not the exact coordinates of a drawing. Our aim is to compute an orthogonal drawing on the grid that has minimum area among all grid drawings that adhere to the given *orthogonal representation.*

This problem is called orthogonal compaction (OC) and is known to be NP-hard, even for orthogonal representations of cycles [1]. We investigate the complexity of OC with respect to several parameters. Among others, we show that OC is fixed-parameter tractable with respect to the most natural of these parameters, namely, the number of *kitty corners* of the orthogonal representation: the presence of pairs of kitty corners in an orthogonal representation makes the OC problem hard. Informally speaking, a pair of kitty corners is a pair of reflex corners of a face that point at each other. Accordingly, the number of kitty corners is the number of corners that are involved in some pair of kitty corners.

References

1 W. S. Evans, K. Fleszar, P. Kindermann, N. Saeedi, C.-S. Shin, and A. Wolff. Mini- mum rectilinear polygons for given angle sequences. Comput. Geom., 100(101820):1– 39, 2022. doi:10.1016/j.comgeo.2021.101820.

4 Open problems

4.1 The Maximum Bimodal Subgraph Problem

Walter Didimo (University of Perugia, IT)

Let G be a plane digraph, i.e., a directed planar graph with a fixed planar embedding. A vertex v of G is bimodal if all its incoming edges (and hence all its outgoing edges) are consecutive in the cyclic order around v. The graph G is bimodal if all its vertices are bimodal. When G is not bimodal, the *maximum bimodal subgraph* problem (MBS) asks to extract from G an embedding preserving subgraph with the maximum number of edges. This problem is known to be NP-hard by Binucci et al. [1], who also describe an efficient heuristic and a branch-and-bound expnential-time algorithm.

The goal is to initiate the study of the MBS problem from the parameterized complexity perspective. In particular, investigate the existance of a polynomial kernel of the MBS parameterized by the number of non-bimodal vertices. Also, is the MBS problem in FPT for some of its structural parameterizatons?

References

1 Carla Binucci, Walter Didimo, and Francesco Giordano. Maximum upward planar subgraphs of embedded planar digraphs. *Comput. Geom.*, 41(3):230–246, 2008.

2 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010.

3 Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. Assoc. Comput. Mach.*, 41(1):153–180, 1994.

4.2 Planar $\mathcal{F}$-Augmentation of Clustered Graphs

Giordano Da Lozzo (University of Rome III, IT) and Siddharth Gupta (University of Warwick – Coventry, GB)

A *flat clustered graph* (for short *flat c-graph*) is a pair $(G, \mathcal{V})$ where G is a graph and $\mathcal{V}$ is a partition of the vertices of G. The parts of $\mathcal{V}$ are called *clusters*. A flat c-graph is *independent* if each of its clusters induces an independent set. Given a family $\mathcal{F}$ of connected graphs and an independent flat c-graph $\mathcal{C} = (G, \mathcal{V} = \{V_1, \ldots, V_k\})$, the PLANAR $\mathcal{F}$-AUGMENTATION problem for $\mathcal{C}$ asks to test for the existence of sets $Z_1, \ldots, Z_k$ of non-edges of G such that $G \cup \bigcup_{i=1}^{k} Z_i$ is planar and $G(V_i, Z_i) \in \mathcal{F}$ for $i = 1, \ldots, k$. If $\mathcal{F}$ is the family of all paths, then the problem coincides with the CLUSTERED PLANARITY WITH LINEAR SATURATORS problem, which is known to be NP-complete [1]. On the other hand, if $\mathcal{F}$ is the family of all trees, then the problem coincides with the CLUSTERED PLANARITY problem, which has been recently shown to be solvable in polynomial time [2].

We highlight some interesting research directions in the study of the PLANAR $\mathcal{F}$-AUGMENTATION problem. First, concerning the setting in which $\mathcal{F}$ is the set of all paths, the mentioned NP-completeness stimulates the search for exact parameterized and sub-exponential algorithms for the problem, both in the fixed and in the variable embedding setting. Moreover, albeit the problem is known to be NP-complete if all clusters, except one, are *trivial*, i.e., they each consist of a single vertex, it is still unknown whether the problem is para-NP-hard in the total number of clusters. Finally, as paths and trees may have linear radius, we regard as an appealing question, at the other side of the radius spectrum, to settle the complexity of the problem when $\mathcal{F}$ is the family of all stars. In this latter setting, it is possible to show that the problem reduces to a constrained planarity problem on the graph obtained by identifying all the vertices belonging to the same cluster, whose study may be of independent interest.

References

1 Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, Ignaz Rutter: *Intersection-Link Representations of Graphs*. J. Graph Algorithms Appl. 21(4): 731-755 (2017)

2 Radoslav Fulek, Csaba D. Tóth: *Atomic Embeddability, Clustered Planarity, and Thickenability*. J. ACM 69(2): 13:1-13:34 (2022)

4.3 The complexity of recognizing Witness Graphs

Eduard Eiben (Royal Holloway, University of London, GB)

Main reference Boris Aronov, Muriel Dulieu, Ferran Hurtado: "Witness (Delaunay) graphs", Comput. Geom., Vol. 44(6-7), pp. 329–344, 2011.
URL https://doi.org//10.1016/j.comgeo.2011.01.001

Witness graphs are a generalization of proximity graphs introduced by Aronov, Dulieu, and Hurtado [1]. A witness graph $G = (V, E)$ is defined by a quadruple $(P, S, W, \pm)$: P is a set of points in plane and $V = P$; S provides the geometric shapes (disks, squares, rectangles); W is set of "witness" points/witnesses in plane; and $uv \in E$ if and only if

- if $+$: some $s \in S$ covers u, v, w for some $w \in W$
- if $-$: some $s \in S$ covers u, v, but no $w \in W$

Examples of witness graphs include:

- Any Delaunay Triangulation can be defined as $(P, S, P, -)$, with S being the set of all disks.
- $(P, S, \emptyset, -)$ defines a complete graph on P as the vertex set.
- Witness Gabriel Graphs [2] are defined by $(P, S, W, -)$, where S contains disks for pairs $u, v \in P$, such that uv is diameter.

Open Questions

What is the complexity of the following problems:

1. Given a graph G and positive integer k, can we represent it as Witness Delaunay Graph/W. Gabriel Graph/Witness Square Graph with at most k witnesses (i.e., $|W| \leq k$)?
2. Given graph G, the set of points P, and S, can we find W such that G is witness graph represented by $(P, S, W, -)$ or $(P, S, W, +)$? A polynomial time algorithm is known for Witness Gabriel Graph. However, the complexity of finding a minimum size W is open even for Witness Gabriel Graph.

References

1 B. Aronov, M. Dulieu, and F. Hurtado. Witness (delaunay) graphs. *Comput. Geom.*, 44(6-7):329–344, 2011.

2 B. Aronov, M. Dulieu, and F. Hurtado. Witness gabriel graphs. *Comput. Geom.*, 46(7):894–908, 2013.

4.4 Depth-first search trees of minimum height

Petr A. Golovach (University of Bergen, NO)

Spanning trees, in particular, depth-first spanning trees (DFS-trees) are popular tools in graph drawing. For these purposes, it may be convenient to consider DFS-trees with additional properties like minimum/maximum number of leaves or minimum height. The additional constraints usually make finding such trees NP-hard. The investigation of the parameterized complexity of problems of this type was initiated by Fellows, Friesen, and Langston [1]. We are interested in the parameterized complexity of Min Height DFS-Tree problem whose task is, given a graph G and an integer h, to decide whether G has a DFS-tree of height at most h. This problem is known to be NP-complete [1]. It can be observed that if G has a DFS-tree of height at most h then the treedepth of G is upper bounded by h. This implies that Min Height DFS-Tree is FPT when parameterized by h. Is it possible to give an efficient, say, single-exponential direct algorithm for the problem? Another direction of research is to investigate the (parameterized) approximability of the minimum height of a DFS-tree.

References

1 M. R. Fellows, D. K. Friesen, and M. A. Langston, *On Finding Optimal and Near-Optimal Lineal Spanning Trees*, Algorithmica, 3 (1988), pp. 549–560.

4.5 Sequential Crossing Number

Thekla Hamm (Utrecht University, NL)

One of the most classically important features of graph drawings is a small number of crossings and the Crossing Number problem, which asks for the minimum number of crossings a graph can be drawn with , is arguably the most famous graph drawing problem. A driving motivation of the field of graph drawing is the visualisation of graphs – a task which is ubiquitous in areas such as the analysis of social networks, bioinformatics and linguistics or generally when information can be captured by graphs. Often such information can evolve over time and hence the treatment of temporal aspects is an important challenge in graph drawing. In particular, is reasonable to require some "stability" of drawing of a graph that changes over time in the sense that parts of the graph occur in consecutive time steps are not redrawn in a different way.

Motivated by this, we define the Sequential Crossing Number problem as generalisation of Crossing Number in which, given a sequence of graphs, we want to draw each of them in a way that drawings of intersections of consecutive graphs agree, and the total number of crossings (i.e. the sum of the number of crossings in each drawing) is minimised.

For $i \in [t-1]$ we say that G_i and G_{i+1} are *consecutive*. Let us also point out two alternative objectives that could be motivated in the same spirit: Instead of requiring $\sum_{i\in[t]} \mathtt{crn}(\mathcal{G}_i) \leq k$, it could also be interesting to require $\max_{i\in[t]} \mathtt{crn}(\mathcal{G}_i) \leq k$, i.e. minimise the maximum number of crossings in any graph, or $\sum_{i\in[t]} \mathtt{crn}(\mathcal{G}_i) - \mathtt{crn}(\mathcal{G}_i|_{G_i\cap G_{i-1}}) \leq k$, i.e. minimise the total number of crossings but counting crossings the drawing of the intersection of consecutive graphs only once.

Of course, classic Crossing Number is captured by $t = 1$, and this already implies NP-hardness of the problem. This is why our open problems will target parameterised algorithms for Sequential Crossing Number. Unless specified otherwise, we consider the parameterisation by k.

Sequential Crossing Number is closely related to the Simultaneous Crossing Number problem [1] which was proposed as a natural generalisation of the well-known Simultaneous Embedding with Fixed Edges problem (t-SEFE for short and most prominently considered for $t = 2$) [2], but has not received a lot of attention since. The conceptual difference between Sequential Crossing Number and Simultaneous Crossing Number is that the former ensures stability of the drawings of subgraphs as long as they appear, whereas the latter additionally ensures stability of drawings of subgraphs that may disappear and reappear over time. t-SEFE is equivalent to Simultaneous Crossing Number for $k = 0$.

Both problems are, by definition equivalent as soon as there is some graph H such that $\forall i \in [t-1]\ G_i \cap G_{i+1} = H$. In particular, this means that the known NP-hardness for t-SEFE on such instances, even if $t = 3$ and H is a star implies that we cannot hope for a parameterised (even XP) algorithm for Sequential Crossing Number in $k + t$, even on instances with a common pairwise intersection of consecutive graphs which is a star.

We pose the general open problem:

▶ **P 1.** *Can we identify conditions on the pairwise intersections of consecutive graphs under which there is an* FPT *or* XP*-algorithm for Sequential Crossing Number in k or $k + t$?*

We propose two concrete directions for such conditions: In the special case that $k = 0$ and each intersection of consecutive graphs is 3-connected, the Sequential Crossing Number is equivalent to testing the planarity of each graph, and hence polynomial-time solvable. Moreover, branching and rigidising the drawings of intersections of consecutive graphs which contain crossings and branching on the graphs whose drawings contain crossings involving edges outside of any consecutive intersection in a hypothetical solution reduces Sequential Crossing Number on instances in which the pairwise intersections of consecutive graphs are 3-connected to Partially Predrawn Crossing Number (the problem of deciding the minimum number of crossings required to extend a drawing of a subgraph to the entire graph). Partially Predrawn Crossing Number is known to be in FPT [3]. From this it can be argued that Sequential Crossing Number on instances in which the pairwise intersections of consecutive graphs are 3-connected is in XP.

▶ **P 2.** *Is Sequential Crossing Number on instances in which the pairwise intersections of consecutive graphs are 3-connected in* FPT*?*

A similar idea would be to rigidise branched drawings for small intersections of consecutive graphs which could yield an XP-algorithm for Sequential Crossing Number parameterised by $k + t + \max_{i\in[t-1]} |E(G_i \cap G_{i+1})|$.

▶ **P 3.** *Is Sequential Crossing Number in* FPT *parameterised by $k+t+\max_{i\in[t-1]} |E(G_i\cap G_{i+1})|$?*

In a different direction polynomial-time algorithm known for t-SEFE on instances with a common pairwise intersection of all input graphs requires that the pairwise intersection of graphs be 2-connected. It would be interesting to extend this result:

▶ **P 4.** *Is Sequential Crossing Number on instances with a common pairwise 2-connected intersection of consecutive graphs in* FPT*? Is this even in* XP*? What about the parameterisation by $(k + t)$?*

References

1 M. Chimani, M. Junger and M. Schulz, "Crossing Minimization meets Simultaneous Drawing", 2008 IEEE Pacific Visualization Symposium, 2008, pp. 33-40, doi: 10.1109/PACIFICVIS.2008.4475456.

2 I. Rutter, "Simultaneous Embedding", Beyond Planar Graphs, 2020, doi: 10.1007/978-981-15-6533-5_13

3 T. Hamm and P. Hliněný, "Parameterised Partially-Predrawn Crossing Number", 38th International Symposium on Computational Geometry (SoCG 2022), 2022, doi: 10.4230/LIPIcs.SoCG.2022.46

4.6 On Left-Aligned BFS Trees

Petr Hlinený (Masaryk University – Brno, CZ)

Main reference Petr Hlinený, Jan Jedelský: "Twin-width of Planar Graphs is at most 8", CoRR, Vol. abs/2210.08620, 2022.
URL https://doi.org//10.48550/arXiv.2210.08620

The concept of a *left-aligned BFS tree* of a planar graph, defined below, was formulated by P. Hliněný while working on an upper bound on the twin-width of planar graphs. It is an important ingredient in the currently best such bound of 8 by Hliněný and Jedelský (briefly noting, without left-aligned BFS trees the same proof would yield only an upper bound of 11). A left-aligned BFS tree can be computed in linear time for a given planar embedding. The featured question (or open problem) is whether the concept of a left-aligned BFS tree can find other applications in algorithmic problems related to planarity, and in particular in the graph drawing area.

▶ **Definition.** *Consider a plane graph G, and a BFS tree T spanning G and rooted in a vertex r of the outer face of G, and picture (for clarity) the embedding G such that r is the vertex of G most at the top. An informal intention is to have a BFS tree T such that there is no other BFS tree of G which is "to the left" of T in at least some place of the geometric picture of G and T. Formally, for two adjacent vertices $u, v \in V(G)$, $\{u, v\} \in E(G)$, we say that u is to the left of v (wrt. T) if neither of u, v lies on the vertical path from r to the other, and the following holds; if r' is the least common ancestor of u and v in T and $P_{r',u}$ (resp., $P_{r',v}$) denote the vertical path from r' to u (resp., v), then the cycle $(P_{r',u} \cup P_{r',v}) + uv$ has the triple (r', u, v) in this counter-clockwise cyclic order. A BFS tree T of G with the BFS layering $\mathcal{L} = (L_0, L_1, \ldots)$ is called* left-aligned *if there is no edge $f = uv$ of G such that, for some index i, $u \in L_{i-1}$ and $v \in L_i$, and u is to the left of v.*

We remark that this concept might seem closely related to the previously used leftist canonical ordering of planar graphs by Badent, Baur, Brandes, and Cornelsen, but it is not that close since the latter does have have the BFS property. Yet, possible applications of both notions may lie in similar areas.

4.7 The *st*-Planar Graph Augmentation Problem

Fabrizio Montecchiani (University of Perugia, IT)

A directed graph G is an st-planar graph if it admits a planar embedding such that: (i) it contains no directed cycle; (ii) it contains a single source vertex s and a single sink vertex t; and (iii) s and t both belong to the outer face of the planar embedding. A popular result in graph drawing states that a directed acyclic graph (DAG) G has an upward planar drawing if and only if G is a subgraph of an st-planar graph. Since testing upward planarity is NP-complete, it follows that testing if a graph is a subgraph of an st-planar graph is also hard. On the other hand, checking if a DAG is st-planar can be easily done in polynomial time.

Based on this motivation we propose to study the st-Planar Graph Augmentation Problem (stPA), which is defined as follows. Let $G = (V, E)$ be a DAG, and let k be a positive integer. Is it possible to *add* at most k edges to G such that the resulting graph is an st-planar graph up to the addition of a super source s and of a super sink t in the outer face of the graph? As a natural open problem, we ask whether stPA admits a fixed-parameter tractable algorithm when parameterized in its natural parameter k.

4.8 The parameterized complexity of Queue Number using structural parameters

Sebastian Ordyniak (University of Leeds, GB)

The Queue Number of an undirected graph G is the minimum number of pages h such that G has s h-queue layout (see also `https://en.wikipedia.org/wiki/Queue_number`).

An h-queue layout of $G = (V, E)$ is an ordering $<_q$ of the vertices together with a partition $\mathcal{P}$ of the edge set of G into h sets such that for every $P \in \mathcal{P}$, if the vertices of G are drawn on the line according to the ordering $<_q$, then no two edges in P form a rainbow, i.e., there are no two edges$e = (u, v)$ and $e = (x, y)$ in P such that $u < x < y < v$. The following is known about Queue Number:

- NP-complete even for one page.
- polynomial-time for fixed ordering (due to k-rainbow obstructions)
- For one page: known to be fpt by treedepth [1]
- FPT by vertex cover for arbitrary number of pages

Open Problems:

- For one page: FPT by treewidth or feedback edge number?
- For arbitrary number of pages: FPT by feedback edge number, treedepth, treewidth?

References

1 Sujoy Bhore, Robert Ganian, Fabrizio Montecchiani and Martin Nöllenburg, *Parameterized Algorithms for Queue Layouts.* J. Graph Algorithms Appl., vol. 26, number 3, pp. 335–352, 2022

4.9 Upward Planarity Testing Parameterized by Treewidth

Kirill Simonov (Hasso-Plattner-Institut, Universität Potsdam, DE)

In Upward Planarity Testing, the input is a directed acyclic graph (DAG) G and the question is whether there exists a planar drawing of G such that all edges are drawn upward, i.e., all edges monotonically increase in the vertical direction. Upward Planarity Testing has been extensively studied, and was shown to be NP-complete already 25 years ago [1], while many polynomial-time algorithms in special cases were also known.

As an outcome of the previous edition of the seminar, it has been shown that Upward Planarity Testing admits an algorithm with running time $n^{O(\mathrm{tw})}$ [2], where n is the number of vertices in the graph and tw is its treewidth. In parameterized complexity terms, Upward Planarity Testing is thus in XP when parameterized by the treewidth of the underlying undirected graph.

A natural question is then whether the algorithmic result above can be improved. Specifically, is Upward Planarity Testing W[1]-hard or FPT parameterized by treewidth? If the former holds, is the running time of $n^{O(\mathrm{tw})}$ tight. For example, can it be shown that there is no $n^{o(\mathrm{tw})}$-time algorithm under the ETH?

References

1 Ashim Garg, Roberto Tamassia: *On the computational complexity of upward and rectilinear planarity testing*. SIAM J. Comput. 31(2):601–625 (2001)

2 Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Robert Ganian, Chrysanthi N. Raftopoulou, Kirill Simonov: *Parameterized Algorithms for Upward Planarity*. SoCG 2022: 26:1–26:16

4.10 Graph embeddings: Drawing vs. learning

Manuel Sorge (TU Wien, AT)

Graph drawing is about finding embeddings of graphs in low-dimensional space. I proposed to look at the embedding problem from a new angle, namely, from a machine-learning point of view. In machine learning, embeddings are also mappings into a low-dimensional space, that represents a condensed form of information, from a high-dimensional space, such as the space of human-written documents or the space of graphs from an application area. The goals of the classical embedding in graph drawing and embedding in machine learning are related, but different: In graph drawing, we usually aim for readability by, e.g., minimizing the number of crossings. In machine learning, we aim to embed similar entities (such as Dagstuhl open problem descriptions) close to each other in the embedding space. A famous embedding algorithm in the machine-learning realm is Node2vec [1], which aims to map the vertex set of a given graph to a low-dimensional embedding space such that similar vertices, as measured by a certain neighborhood relation based on co-occurrence in random walks, are close to each other in the embedding space.

It seems interesting to:

1. Explore machine-learning embedding problems from a visualization point of view, i.e., is a graph easier to understand if we embed similar vertices close to each other in 2d-space, like in Node2vec?
2. Consider discrete variants of embedding problems: The dimensions in the embedding space can be thought of as underlying features of the data. For interpretability, it would be desirable to have few easy-to-understand features. I.e., it would be interesting to map to low-dimensional spaces where each dimension has only small domain, that is, a small number of admissible values.
3. Develop efficient algorithms that exploit properties such as the low dimensionality of the embedding space, small domains, and input graph structure.

For a concrete starting point, we may look at a simplified version of Node2vec: Here, we are given a graph $G = (V, E)$ and we aim to find a mapping $f\colon V \to \mathbb{R}^d$ for some small d such that $\sum_{u \in V} \log \Pr(N(u) \mid f(u))$ is maximized. Intuitively and simplified, we want to maximize the number of vertices with the same neighborhood that are mapped to the same point in the embedding space. After several approximations, Grover and Leskovec [1] arrive at the objective to maximize

$$\sum_{u \in V} \left[-\log(Z_u) + \sum_{v \in N_S(u)} f(u)f(v) \right],$$

where $Z_u = \sum_{v \in V} \exp(f(u)f(v))$. Herein, we can think of the logarithmic term as a baseline and each term in the sum measures the deviation from the baseline. Simplifying further, we may observe that in the vertices in the baseline term are weighed lower (due to taking the logarithm) than the vertices in the other term. If we weigh both at exactly half instead, we obtain the following objective:

$$\sum_{u \in V} \left[\sum_{v \in N(u)} (f(u)f(v)) - \sum_{v \in V \setminus N(u)} f(u)f(v) \right].$$

For instance, if we simply want to find an embedding into binary, one-dimensional space $\{0, 1\}$, we thus want to find an induced subgraph (corresponding to the vertices assigned 1) that maximizes the number of edges contained in the subgraph minus the number of nonedges in the subgraph. A natural first step is to find some baseline complexity results or promising algorithms by looking at such special cases and then generalize to larger domains and dimensions.

References

1 Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864. ACM, 2016. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939754. URL `https://doi.org/10.1145/2939672.2939754`.

4.11 Drawing Hypergraphs "Nicely"

Yushi Uno (Osaka Metropolitan University, JP)

Hypergraphs [1] are not only theoretically interesting as a generalized notion of graphs, but also has a wide range of applications in many fields, such as database theory, biology, and so on [3]. Therefore, drawing and visualizing hypergraphs is useful for understanding practical cases in applied fields.

Examples of such visualized hypergraphs have been appeared in typical textbooks or papers [1, 2], however, there are many different ways to draw them, and a unified approach has not yet been fully established. On the other hand, it is often experienced that the level of understanding a hypergraph varies greatly depending on how it is drawn, and there is a need for "nice" drawing of hypergraphs.

The criteria for goodness or understandability of a hypergraph drawing could simply be the number of crossings of hyperedges or the number of curved lines, for example, but the definition of the crossings itself is not necessarily trivial that might be depending on the drawing model. Thus, for example, the following kind of problem can be considered:

Minimum Number of Curved Lines
Input: a k-uniform hypergraph ($k \geq 3$),
Output: under "straight and curved line model", what is the minimum number of curved lines, and what is its node layout?

We would then like to discuss drawing algorithms that optimize such criteria and analyze these algorithms in terms of their parameterized computational complexity. Possible parameters include the number of hyperedges, the number of crossings, the maximum degree, uniformity, and so on.

This topic is closely related to the notion of set visualization, e.g., `https://www.dagstuhl.de/en/seminars/seminar-calendar/seminar-details/22462`, so we have to scrutinize the results in that field.

References

1 C. Berge. *Graphs and Hypergraphs.* North-Holland, 1973.
2 V. I. Voloshin. *Introduction to Graph and Hypergraph Theory.* Nova Science, 2009.
3 R. Fagin. Degrees of acyclicity for hypergraphs and relational database schemes. *J. ACM*, 30(3), 514–550, 1983.

5 Working groups

5.1 A Subexponential FPT Algorithm for Deleting Few Edges from Ordered Graphs to Make Them p-Plane

Akanksha Agrawal (Indian Institute of Techology Madras, IN), Sergio Cabello (University of Ljubljana, SI), Michael Kaufmann (Universität Tübingen, DE), Saket Saurabh (The Institute of Mathematical Sciences – Chennai, IN), Roohani Sharma (MPI für Informatik – Saarbrücken, DE), Yushi Uno (Osaka Metropolitan University, JP), and Alexander Wolff (Universität Würzburg, DE)

Introduction

Since (many) crossings usually make it hard to understand the drawing of a graph, much effort in the area of Graph Drawing has been directed towards reducing the number of crossings in drawings of graphs. In terms of parametrized complexity, several facets of this problem have been considered. For example, there are FPT algorithms that, given a graph G and an integer k, decide whether G can be drawn with at most k crossings [8, 12].

Crossing minimization has also been considered in the setting where each vertex of the given graph must lie on one of two horizontal lines. This restricted version of crossing minimization is an important subproblem in drawing layered graphs according to the so-called *Sugiyama framework* [17]. There are two variants of the problem; either the vertices on both lines may be freely permuted or the order of the vertices on one line is given. These variants are called two-layer and one-layer crossing minimization, respectively. For both, FPT algorithms exist [5, 13].

Surprisingly, crossing minimization remains NP-hard even when restricted to graphs that have a planar subgraph with just one edge less [2]. Another way to deal with crossings is to remove a small number of vertices or edges such that the remaining graph can be drawn without crossings. In fact, it is known that vertex deletion to planarity is FPT with respect to the number of deleted vertices [9, 11, 15]. However, the running times of these algorithms depends at least exponentially on the number of deleted vertices. On the kernelization front, there exists an α-approximate kernel for vertex deletion to planarity [10], whereas vertex deletion to outerplanarity is known to admit an (exact) polynomial kernel [4].

In the following, we consider a restricted setting that allows us to obtain substantially better running times although the graphs to which we delete are more complex than (outer-)planar graphs.

An *ordered graph* (G, σ) consists of a graph G and an ordering $\sigma\colon V(G) \to \{1, \dots, |V(G)|\}$ of the vertices of G. For two edges $e = (u, v)$ and $e' = (u', v')$ of an ordered graph (G, σ), we say that e and e' *cross with respect to* σ if $\sigma(u) < \sigma(u') < \sigma(v) < \sigma(v')$ or if $\sigma(u') < \sigma(u) < \sigma(v') < \sigma(v)$. Let $\texttt{cross}_{(G,\sigma)}(e)$ denote the set of edges of G that cross e with respect to σ. We drop the subscript (G, σ) when it is clear from the context. The ordered graph models the scenario where the vertices of G are placed along a horizontal line in the order given by σ and all the edges are drawn above the line using curves that cross as few times as possible. Whenever e and e' cross with respect to σ, their curves must intersect. Whenever e and e' do *not* cross with respect to σ, their curves can be drawn without intersections; for example, we may use halfcircles. In this setting, we get a drawing such that two edges of G cross precisely if and only if they cross with respect to σ.

We consider the edges of G as directed from the vertex with smaller index to that of larger index, that is, edge uv is directed (u, v) if $\sigma(u) < \sigma(v)$. Given a positive integer p, we say that an ordered graph (G, σ) is p-plane if every edge in G is crossed by at most p other edges. We study the parameterized complexity of the following problem.

Ordered Graph Edge Deletion to p-Plane **Parameter:** k
Input: An ordered graph (G, σ), and a positive integer k
Question: Does there exist a set $S \subseteq E(G)$ of at most k edges such that $(G - S, \sigma)$ is p-plane.

An instance of this problem is denoted $((G, \sigma), k)$ or simply (G, σ, k).

Given an ordered graph (G, σ), define the *conflict graph* $H_{(G,\sigma)}$ as the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G. Using this notion, we can express Ordered Graph Edge Deletion to p-Plane equivalently as the problem of deleting from $H_{(G,\sigma)}$ a set of at most k vertices such that the remaining graph has maximum degree at most p. For general graphs, this problem is called Vertex Deletion to Degree-p [16]; it admits a quadratic kernel [7, 18].

A Fast FPT Algorithm

The main result of this report is as follows.

▶ **Theorem 1.** *Ordered Graph Edge Deletion to p-Plane admits an algorithm with running time $2^{\mathcal{O}(p\sqrt{k}\log(p+k))} \cdot n^{\mathcal{O}(1)}$.*

In other words, we obtain a subexponential fixed-parameter tractable algorithm parameterized for Ordered Graph Edge Deletion to p-Plane parameterized by k; note that we consider p to be a constant here (although we made explicit how the running time depends on p). Our algorithm to prove Theorem 1 has two steps. First it branches on edges that are crossed by at least $p + \sqrt{k}$ other edges. When such edges do not exist, we show that the conflict graph $H_{(G,\sigma)}$ has treewidth $O(p + \sqrt{k})$, which allows us to use a known (folklore) algorithm [14] for Vertex Deletion to Degree-p whose dependency is singly exponential in the treewidth of $H_{(G,\sigma)}$.

Branching

We show that we can use branching to reduce any instance to a collection of instances where each edge e of the graph satisfies $|\texttt{cross}(e)| < p + \sqrt{k}$.

Let e be an edge of G with $|\texttt{cross}(e)| \geq p + \lceil\sqrt{k}\rceil$. If $|\texttt{cross}(e)| > p + k$, then e must be deleted, as we cannot afford to keep e and delete enough edges from $\texttt{cross}(e)$. If $|\texttt{cross}(e)| \leq p + k$, then either e must be deleted or at least $|\texttt{cross}(e)| - p$ many edges from $\texttt{cross}(e)$ must be deleted, so that at most p edges of $\texttt{cross}(e)$ stay. This results in the following branching rule, where we return an OR over the answers of the following instances:

1. Recursively solve the instance $(G - e, \sigma, k - 1)$. This branch is called the *light* branch.
2. If $|\texttt{cross}(e)| > p + k$, we do not consider other branches. Otherwise, for each subset X of $\texttt{cross}(e)$ with $|\texttt{cross}(e)| - p$ many edges, recursively solve the instance $(G - X, \sigma, k - |X|)$. Each of these branches are called *heavy* branches.

We are going to show that the recursion tree has $2^{O(p\sqrt{k}\log(p+k))}$ branches. Note that we have

$$\binom{|\texttt{cross}(e)|}{|\texttt{cross}(e)|-p} \leq \binom{|\texttt{cross}(e)|}{p} \leq \binom{p+k}{p} \leq (p+k)^p$$

possible heavy branches at each node. Let $\kappa = (p+k)^p$.

To prove the desired upper bound we interpret the branching tree as follows. First note that, in each node, we have at most κ heavy branches. We associate a distinct word over the alphabet $\Sigma = \{0, 1, \ldots, \kappa\}$ to each leaf (root to leaf path) of the recurrence tree. For each node of the recurrence tree, associate a character from Σ in each of its children such that the child node corresponding to the light branch gets appended the character 0 and the other nodes (corresponding to the heavy branches) get a distinct character from Σ. Now a word over the alphabet Σ for a leaf ℓ of the recurrence tree is obtained by taking the sequence of character on the nodes of the root to leaf ℓ, in order. In order to bound the number of leaves (and hence the total number of nodes) of the recurrence tree, it is enough to bound the number of such words. The character 0 is called a *light* label and all other characters are called *heavy* labels. Recall that a light label corresponds to the branch where k drops by 1, while the heavy labels correspond to the branches where k drops by $|\texttt{cross}(e)| - p \geq \sqrt{k}$. This implies that each word (that is associated with the leaf of the recurrence tree) has at most $\sqrt{k}$ heavy labels. In order to bound the number of such words, we first guess the places in the word that are occupied by heavy labels and then we guess the (heavy) labels itself at these selected places. All other positions have the light label on them and there is no choice left. Hence, the number of such words is upper bounded by

$$\sum_{i=0}^{\sqrt{k}} \binom{k}{i}\kappa^i \leq \sqrt{k}\binom{k}{\sqrt{k}}\kappa^{\sqrt{k}} \leq \sqrt{k}\binom{k}{\sqrt{k}}((p+k)^p)^{\sqrt{k}} = 2^{\mathcal{O}(p\sqrt{k}\log(p+k))}.$$

This shows that the number of such words are upper bounded by $2^{\mathcal{O}(p\sqrt{k}\log(p+k))}$, and hence the number of leaves (and nodes) of the recurrence tree are upper bounded by $2^{\mathcal{O}(p\sqrt{k}\log(p+k))}$.

Balanced Separators

Let (G, σ) be an ordered graph. For any edge $e = (u, v)$ of G, let $\texttt{span}_{(G,\sigma)}(e)$ be the set of all edges $(u', v') \neq e$ of G such that $\sigma(u) \leq \sigma(u') \leq \sigma(v') \leq \sigma(v)$. For any vertex w of G, let $\texttt{left}_{(G,\sigma)}(w)$ be the set of all edges (u, v) of G such that $\sigma(u) < \sigma(v) \leq \sigma(w)$. Whenever it is clear from the context, we will drop the subscript (G, σ). We say that an edge e of G is *maximal* if G contains no edge e' such that $e \in \texttt{span}(e')$,

▶ **Lemma 2** (Balanced Separator). *If (G, σ) is an ordered q-plane graph, then G contains a set X of at most $2(q+1)$ edges such that $E(G) \setminus X = E_1 \uplus E_2$, $|E_1| \leq 2m/3$, $|E_2| \leq 2m/3$, and no edge $e_1 \in E_1$ crosses an edge $e_2 \in E_2$ with respect to σ.*

Proof. We consider three cases depending on the spans of the edges of G.

Case 1: There exists an edge $e = (u, v) \in E(G)$ such that $m/3 \leq |\texttt{span}(e)| \leq 2m/3$.

In this case, let $X = \texttt{cross}(e) \cup \{e\}$, let $E_1 = \texttt{span}(e)$, and let $E_2 = E(G - X) \setminus E_1$. Note that $|X| \leq q + 1$, $|E_1| \leq 2m/3$, and $|E_2| \leq 2m/3$. Now let $e_1 = (u_1, v_1) \in E_1$ and

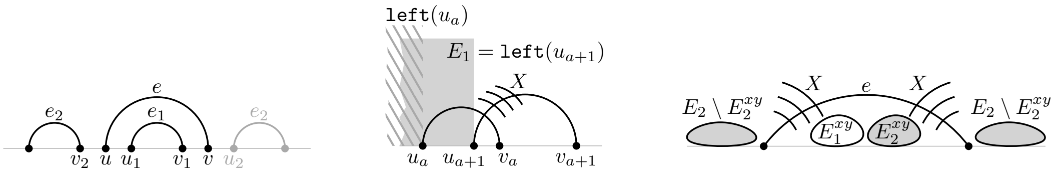

Figure 1 Case distinction for the proof of theorem 2.

$e_2 = (u_2, v_2) \in E_2$. Since $e_1 \in E_1$, we have $\sigma(u) \leq \sigma(u_1) < \sigma(u_2) \leq \sigma(v)$; see Case 1, fig. 1. Since $e_2 \in E_2$, we have $\sigma(v_2) < \sigma(u)$ or $\sigma(v) < \sigma(u_2)$; see the black and the gray version of e_2 in Case 1, fig. 1, respectively. In both cases, e_1 and e_2 do not cross.

Case 2: For every edge $e \in E(G)$, it holds that $|\texttt{span}(e)| \leq m/3$.

Let $M \subseteq E(G)$ be the collection of all maximal edges of G in σ. Let $\mu = |M|$, and let $M = \{(u_1, v_1), \ldots, (u_\mu, v_\mu)\}$, where $\sigma(u_1) < \sigma(u_2) < \cdots < \sigma(u_\mu)$. Note that $|\texttt{left}(u_1)| \leq \cdots \leq |\texttt{left}(u_\mu)|$ and that $|\texttt{left}(u_1)| = 0$. The equality is due to the fact that u_1 is the first non-isolated vertex of G in σ (and v_1 is the rightmost neighbor of u_1).

Let $a \in [p]$ be the largest index such that $|\texttt{left}(u_a)| \leq m/3$. Since $|\texttt{left}(u_1)| = 0$, it is clear that such an index a exists.

We claim that $m/3 < |\texttt{left}(u_{a+1})| \leq 2m/3 + q$. From the choice of a, it is clear that $|\texttt{left}(u_{a+1})| > m/3$. Note that $\texttt{left}(u_{a+1}) \subseteq \texttt{left}(u_a) \cup \texttt{span}((u_a, v_a)) \cup \texttt{cross}((u_a, v_a))$; see Case 2, fig. 1. This yields our claim since $|\texttt{left}(u_{a+1})| \leq |\texttt{left}(u_a)| + |\texttt{span}((u_a, v_a))| + |\texttt{cross}((u_a, v_a))| \leq 2m/3 + q$.

Now let $X = \texttt{cross}((u_{a+1}, v_{a+1})) \cup \{(u_{a+1}, v_{a+1})\}$, $E_1 = \texttt{left}(u_{a+1})$ and $E_2 = E(G - X) \setminus E_1$. Since $m/3 \leq |\texttt{left}(u_{a+1})| \leq 2m/3 + p$, $|E_1| \leq 2m/3 + p$, and $|E_2| \leq 2m/3$. Finally, we simply move p edges from E_1 to X. Then $|X| \leq 2p + 1$ and $|E_1| \leq 2m/3$. Given our construction, it is clear that no edge in E_1 crosses any edge in E_2; see Case 2, fig. 1.

Case 3: There exists an edge $e^* \in E(G)$ such that $|\texttt{span}(e^*)| > 2m/3$.

Let $e = (u, v)$ be an edge of G such that $|\texttt{span}(e)| > 2m/3$, and there is no $e' \in |\texttt{span}(e)|$ such that $|\texttt{span}(e')| > 2m/3$. Since Case 1 does not apply, for each $e' \in \texttt{span}(e)$, we have $|\texttt{span}(e')| \leq m/3$. Let $V_{uv} = \{w \in V(G) \colon \sigma(u) \leq \sigma(w) \leq \sigma(v)\}$. Let $G_{uv} = G[V_{uv}]$, and let σ_{uv} be the restriction of σ to V_{uv}.

Now we apply Case 2 to the ordered graph (G_{uv}, σ_{uv}). This yields a set $X_{uv} \subseteq E(G_{uv})$ of size at most $q + 1$, and edge sets E_1^{uv} and E_2^{uv} such that $E(G_{uv}) \setminus X_{uv} = E_1^{uv} \uplus E_2^{uv}$, $m/3 \leq |E_1^{uv}| \leq 2m/3$, $|E_2^{uv}| \leq 2m/3$, and no edge in E_1^{uv} crosses any edge in E_2^{uv}.

Let $X = X_{uv} \cup \texttt{cross}(e) \cup \{e\}$. Then $|X| \leq 2(q + 1)$. Let $E_1 = E_1^{uv}$ and $E_2 = E(G - X) \setminus E_1^{uv}$. Since $m/3 \leq |E_1^{uv}| \leq 2m/3$, clearly $|E_2| \leq 2m/3$. It remains to show that no edge of E_2 crosses any edge of E_1; see Case 3, fig. 1. By construction, no edge of E_2^{uv} crosses any edge of E_1^{uv}. The edges in $E_2 \setminus E_2^{uv}$ neither cross e nor do they lie in $\texttt{span}(e)$, so they cannot cross any edge in $E_1 \subseteq \texttt{span}(e)$. ◀

Proof of Theorem 1. We now need to establish a relation between the treewidth of the graph and the size of a balanced separator in it. For this we use the result of Dvořák and Norin [6] that shows a linear dependence between the treewidth and the *separation number* of a graph: the separation number of a graph is the smallest integer s such that every subgraph of the given graph has a balanced separator of size at most s. In other words, they show that if the separation number of the graph is s, then the treewidth of such a graph is $\mathcal{O}(s)$.

Recall that (G, σ, k) is an instance of Ordered Graph Edge Deletion to p-Plane. By Lemma 2, if the ordered graph (G, σ) is $(p + \sqrt{k})$-plane, then the conflict graph $H_{(G,\sigma)}$ has a balanced separator of size at most $2(p + \sqrt{k} + 1)$. Thus, due to the result of Dvořák and Norin [6], the treewidth of $H_{(G,\sigma)}$ is $\mathcal{O}(p + \sqrt{k})$.

Given a graph with N vertices and treewidth t, one can compute, in $(d + 2)^t \cdot N^{\mathcal{O}(1)}$ time, the smallest set of vertices whose deletion results in a graph of degree at most d [14, Folklore]. Applying this result to the conflict graph $H_{(G,\sigma)}$, which has at most $|V(G)|^2 = n^2$ vertices and treewidth $\mathcal{O}(p + \sqrt{k})$, we conclude that Ordered Graph Edge Deletion to p-Plane can be solved in $2^{\mathcal{O}((p+\sqrt{k}) \log p)} \cdot n^{\mathcal{O}(1)}$ time if the given ordered graph (G, σ) is $(p + \sqrt{k})$-plane.

From Branching, we can assume, at the expense of a multiplicative factor of $2^{\mathcal{O}(p\sqrt{k} \log(k+p))}$ on the running time, that the given ordered graph (G, σ) is $(p + \sqrt{k})$-plane. Thus, given (G, σ, k), we can solve Ordered Graph Edge Deletion to p-Plane in $2^{\mathcal{O}(p\sqrt{k} \log(p+k))}$ time. This concludes the proof of Theorem 1. ◀

Open Problems

1. Could we use the concept of the conflict graph for other crossing reduction problems?
2. Is Ordered Graph Edge Deletion to p-Plane $W[1]$-hard with respect to the natural parameter k if p is part of the input? Can we reduce from Independent Set? Note that Vertex Deletion to Degree-p is $W[1]$-hard with respect to treewidth [1] and that outer-p planar graphs have treewidth $O(p)$ [3] (which also follows from theorem 2).
3. What if the vertex order is not given? In other words, what is the parametrized complexity of edge deletion to outer-p planarity?
4. What about computing the crossing number of an ordered graph? Of course, the total number of crossings is always at least the number of crossings per edge.

References

1 N. Betzler, R. Bredereck, R. Niedermeier, and J. Uhlmann. On bounded-degree vertex deletion parameterized by treewidth. *Discrete Appl. Math.*, 160(1):53–60, 2012. doi:j.dam.2011.08.013.

2 S. Cabello and B. Mohar. Adding one edge to planar graphs makes crossing number and 1-planarity hard. *SIAM J. Comput.*, 42(5):1803–1829, 2013. doi:10.1137/120872310.

3 S. Chaplick, M. Kryven, G. Liotta, A. Löffler, and A. Wolff. Beyond outerplanarity. In F. Frati and K.-L. Ma, editors, *Proc. 25th Int. Symp. Graph Drawing & Network Vis. (GD'17)*, volume 10692 of *LNCS*, pages 546–559. Springer, 2018. URL: https://arxiv.org/abs/1708.08723, doi:10.1007/978-3-319-73915-1_42.

4 H. Donkers, B. M. P. Jansen, and M. Włodarczyk. Preprocessing for outerplanar vertex deletion: An elementary kernel of quartic size. *Algorithmica*, 84(11):3407–3458, 2022. doi:10.1007/s00453-022-00984-2.

5 V. Dujmović and S. Whitesides. An efficient fixed parameter tractable algorithm for 1-sided crossing minimization. *Algorithmica*, 40(1):15–31, 2004. doi:10.1007/s00453-004-1093-2.

6 Z. Dvořák and S. Norin. Treewidth of graphs with balanced separations. *J. Comb. Theory, Ser. B*, 137:137–144, 2019. doi:10.1016/j.jctb.2018.12.007.

7 M. R. Fellows, J. Guo, H. Moser, and R. Niedermeier. A generalization of Nemhauser and Trotter's local optimization theorem. *J. Comput. Syst. Sci.*, 77(6):1141–1158, 2011. doi:10.1016/j.jcss.2010.12.001.

8 M. Grohe. Computing crossing numbers in quadratic time. *J. Comput. Syst. Sci.*, 68(2):285–302, 2004. doi:10.1016/j.jcss.2003.07.008.

9 B. M. P. Jansen, D. Lokshtanov, and S. Saurabh. A near-optimal planarization algorithm. In C. Chekuri, editor, *Proc. Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, pages 1802–1811, 2014. doi:10.1137/1.9781611973402.130.

10 B. M. P. Jansen and M. Włodarczyk. Lossy planarization: a constant-factor approximate kernelization for planar vertex deletion. In S. Leonardi and A. Gupta, editors, *Proc. Ann. ACM SIGACT Symp. Theory Comput. (STOC)*, pages 900–913, 2022. doi:10.1145/3519935.3520021.

11 K. Kawarabayashi. Planarity allowing few error vertices in linear time. In *Proc. Ann. IEEE Symp. Foundat. Comput. Sci. (FOCS)*, pages 639–648, 2009. doi:10.1109/FOCS.2009.45.

12 K. Kawarabayashi and B. A. Reed. Computing crossing number in linear time. In D. S. Johnson and U. Feige, editors, *Proc. 39th Ann. ACM Symp. Theory Comput. (STOC)*, pages 382–390, 2007. doi:10.1145/1250790.1250848.

13 Y. Kobayashi and H. Tamaki. A faster fixed parameter algorithm for two-layer crossing minimization. *Inform. Process. Lett.*, 116(9):547–549, 2016. doi:j.ipl.2016.04.012.

14 M. Lampis and M. Vasilakis. Structural parameterizations for two bounded degree problems revisited. *CoRR*, abs/2304.14724, 2023. doi:10.48550/arXiv.2304.14724.

15 D. Marx and I. Schlotter. Obtaining a planar graph by vertex deletion. *Algorithmica*, 62(3-4):807–822, 2012. doi:10.1007/s00453-010-9484-z.

16 N. Nishimura, P. Ragde, and D. M. Thilikos. Fast fixed-parameter tractable algorithms for nontrivial generalizations of vertex cover. *Discret. Appl. Math.*, 152(1-3):229–245, 2005. doi:10.1016/j.dam.2005.02.029.

17 K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybernetics*, 11(2):109–125, 1981. doi:10.1109/TSMC.1981.4308636.

18 M. Xiao. On a generalization of Nemhauser and Trotter's local optimization theorem. *J. Comput. Syst. Sci.*, 84:97–106, 2017. doi:10.1016/j.jcss.2016.08.003.

5.2 Clustered Planarity

Giordano Da Lozzo (University of Rome III, IT), Robert Ganian (TU Wien, AT), Siddharth Gupta (University of Warwick – Coventry, GB), Bojan Mohar (Simon Fraser University – Burnaby, CA), Sebastian Ordyniak (University of Leeds, GB), and Meirav Zehavi (Ben Gurion University – Beer Sheva, IL)

A *flat clustered graph* (for short *flat c-graph*) is a pair $(G, \mathcal{V})$ where G is a graph and $\mathcal{V}$ is a partition of the vertices of G. The parts of $\mathcal{V}$ are called *clusters*. A flat c-graph is *independent* if each of its clusters induces an independent set. Given a family $\mathcal{F}$ of connected graphs and an independent flat c-graph $\mathcal{C} = (G, \mathcal{V} = \{V_1, \dots, V_k\})$, the Planar $\mathcal{F}$-augmentation problem for $\mathcal{C}$ asks to test for the existence of sets Z_i of edges joining pairs of vertices in V_i for $i = 1, \dots, k$, such that $G \cup \bigcup_{i=1}^{k} Z_i$ is planar and $G(V_i, Z_i) \in \mathcal{F}$ for $i = 1, \dots, k$. If $\mathcal{F}$ is the family of all paths, then the problem coincides with the Clustered Planarity with Linear Saturators problem, which is known to be NP-complete [1]. On the other hand, if $\mathcal{F}$ is the family of all trees, then the problem coincides with the Clustered Planarity problem, which has been recently shown to be solvable in polynomial time [2].

We consider Planar $\mathcal{F}$-augmentation for two classes $\mathcal{F}$, i.e., the set $\mathcal{P}$ of all paths and the set $\mathcal{S}$ of all stars. Since it is known that Planar $\mathcal{P}$-augmentation is NP-hard even if the input graph has no edges, but has many non-trivial clusters (clusters containing more than one vertex), we tried to obtain an fpt-algorithm for the parameter treewidth plus the number of non-trivial clusters using SPQR-trees. Unfortunately, after some consideration, we had to abandon this approach, after realizing that we were not able to handle spiralling paths in the solution; see also Figure 4b) in [3] giving a worst-case example for spiralling paths.

We then focused on obtaining exact algorithms for Planar $\mathcal{P}$-augmentation and we are now pretty confident that we can obtain a single-expoential $2^{O(n)}$ algorithm for the problem in the variable embedding case as well as a subexponential $2^{O(\log n\sqrt{n})}$ algorithm in the fixed embedding case (if the initial graph is connected). Both algorithms start by guessing a separator of the solution graph. Then, in the variable embedding case we additionally guess the partition of the separation (which dominates the run-time in this case and is the reason that we do not obtain a subexponential-time algorithm). After doing so the algorithm proceeds recursively in both parts. In the fixed embedding and connected case, we can circumvent guessing the partition, by guessing the curve through the separator, which leaves only $n^{\sqrt{n}} = 2^{O(\log n\sqrt{n})}$ many choices. We leave it open whether either the variable-embedding case can be improved to a subexponential-time algorithm and whether we can avoid the $\log n$ factor in the exponent for the fixed-embedding case.

Our second main result concerns the Planar $\mathcal{S}$-augmentation problem. Here, we can show that if the input graph is bi-connected, then Planar $\mathcal{S}$-augmentation can be solved in polynomial-time in the variable-embedding case. The main ideas behind this algorithm are as follows. Consider the graph H obtained from G after contradicting every cluster into a single vertex such that every half-edge in H is assigned the color that is equal to the vertex in the cluster that it originated from. Our first observation is that G has a solution if and only if H has a drawing such that all but one color of the half-edges around every vertex is consecutive. In other words there is at most one color that is split at every vertex. We now employ a dynamic programming algorithm along an SPQR-tree of H to decide whether H admits such a drawing. To do so we define the following records for every pertinent graph corresponding to an artificial arc st: a record stores the colors of the two outermost half-edges incident to s, the colors of the two outermost half-edges incident to t, and the color that is split (if there is no split color this remains undefined). It is important to note that every pertinent graph st has at most 3 colors that also occur outside (otherwise at least two colors are split and there is no solution for H) and that we do not need to know the identity of any color that does not occur outside. Therefore, the number of possible colors that we store for every entry of the record is at most 4, i.e., the color could be one of the at most 3 colors that also occur outside or it could be another color (which we will denote with the symbol u). Therefore, the number of records for a pertinent graph is at most $4^4 + 5$ (4 options for the colors of the outer half-edges and 5 options for the split color). It is then relatively straightforward to show how the records are computed for S-nodes. For P-nodes, we design a slightly more involved greedy algorithm that guesses the colors of the outer half-edges and then proceeds by computing the forced implications. Rather more complicated is the algorithm for computing R-nodes. Here, we manage to obtain an encoding into 2-SAT after some preprocessing of the instance. It is not clear to us how to get around the condition that G has to be bi-connected. Neither do we know, whether a similar approach can be used for the fixed-embedding case. The main problem with using our approach for the fixed embedding case is that our initial observation about contradicting every cluster into one vertex does not longer preserve solutions. This is because the color that is split at any vertex depends on, where the contracted vertex is placed within the drawing.

References

1 Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, Ignaz Rutter: *Intersection-Link Representations of Graphs*. J. Graph Algorithms Appl. 21(4): 731-755 (2017)

2 Radoslav Fulek, Csaba D. Tóth: *Atomic Embeddability, Clustered Planarity, and Thickenability*. J. ACM 69(2): 13:1-13:34 (2022)

3 Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani: *2-Level Quasi-Planarity or How Caterpillars Climb (SPQR-)Trees*. CoRR: abs/2011.02431 (2020)

5.3 The Maximum Bimodal Subgraph Problem: Parameterized Complexity and Approximation Scheme

Walter Didimo (University of Perugia, IT), Fedor V. Fomin (University of Bergen, NO), Petr A. Golovach (University of Bergen, NO), Tanmay Inamdar (University of Bergen, NO), Stephen G. Kobourov (University of Arizona – Tucson, US), and Marie Diana Sieper (Universität Würzburg, DE)

A *plane digraph* G is a directed planar graph with a given planar embedding. In particular, the embedding of G defines the circular ordering of the edges incident to each vertex. A vertex v of G is *bimodal* if all its incoming edges (and hence all its outgoing edges) are consecutive in the cyclic order around v. A plane digraph is *bimodal* if all its vertices are bimodal. For example, the plane digraph of Figure 2(a) is not bimodal, namely the vertices $3, 4, 5, 7$ are bimodal while the vertices $1, 2, 6$ are not.

Bimodality is a central property for many graph drawing styles. In particular, it is a necessary condition for the existence of *level-planar* and *upward planar* drawings, where the edges are represented as curves monotonically increasing in the upward direction according to their orientations [11, 12, 13, 18]. Bimodality becomes also a sufficient condition for the existence of embedding-preserving *quasi-upward planar* drawings, in which edges are allowed to violate the upward monotonicity a finite number of times at points called *bends* [5, 6, 7]. Recently, it has been shown that bimodality is also sufficient for the existence of *planar L-drawings* of digraphs, in which distinct L-shaped edges may overlap but not cross [1, 2, 3].

Given a digraph G it is possible to decide in linear time if G admits a bimodal planar embedding, by simply transforming the problem into a planarity testing problem on a suitable digraph obtained by splitting each non-bimodal vertex of G into two vertices [5]. However, when G is not bimodal, a problem that naturally arises is to extract from G a subgraph of maximum size (i.e., with the maximum number of edges) that is bimodal. Unfortunately, this problem is known to be NP-hard, even when G has a given planar embedding and we look for an embedding-preserving maximum bimodal subgraph [8]. This problem is called the *Maximum Bimodal Subgraph* (MBS) problem. For example, Figure 2(b) shows a bimodal subgraph of maximum size for the graph of Figure 2(a).

Binucci et al. [8] describe a heuristic and an exponential-time branch-and-bound algorithm to solve MBS, by also addressing the more specific problem of extracting from the input plane digraph G an upward planar digraph of maximum size.

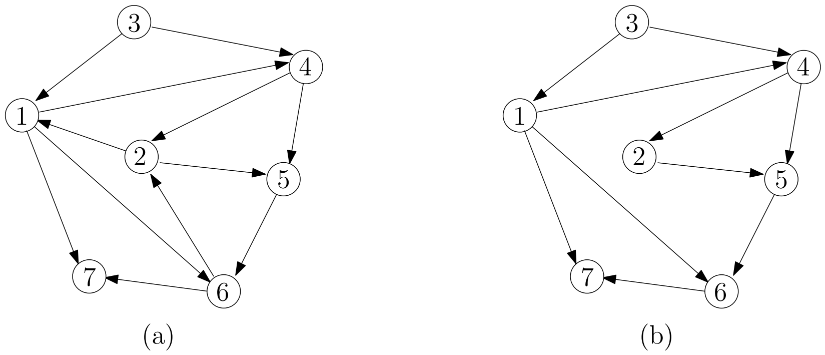

Figure 2 (a) A plane digraph G. (b) A maximum bimodal subgraph of G that preserves its embedding.

Contribution

During the Dagstuhl Seminar we studied the MBS problem from the parameterized complexity and approximability perspectives (we refer to the books [10, 15] for an introduction to the parameterized complexity area). More precisely, we considered the following more general version of the problem with weighted edges, which coincides with MBS if we restrict to unit edge weights.

MWBS(G, w) (*Maximum Weighted Bimodal Subgraph*). *Given a plane digraph G and an edge-weight function $w : E(G) \to \mathbb{Q}^+$, compute a bimodal subgraph of G of maximum weight, i.e., whose sum of the edge weights is maximum over all bimodal subgraphs of G.*

We studied different parameters for the MWBS(G, w) problem and mainly focused on the following: (i) the number b of non-bimodal vertices of the input graph G, which intrinsically captures the nature and the difficulty of the problem; (ii) the branchwidth and the treewidth of the undirected underlying graph of G, which are structural parameters commonly adopted in parameterized complexity. Our main results can be summarized as follows.

- *Structural parameterization.* We show that MWBS is FPT when parameterized by the branchwidth of the input digraph G or, equivalently, by the treewidth of G, which is correlated to the branchwidth by a constant factor [19]. Our algorithm deviates from a standard dynamic approach for graphs of bounded treewidth. Since we have to incorporate the "topological" information about the given embedding in the dynamic program, we exploit the sphere-cut decomposition of Dorn et al. [14].
- *Kernelization.* We prove the existence of a polynomial kernel for the decision version of MWBS parameterized by the number b of non-bimodal vertices. Our kernelization consists of several steps. First we show how to reduce the instance to an equivalent instance whose branchwidth is $O(\sqrt{b})$. Second, by using specific gadgets, we compress the problem to an instance of another problem whose size is bounded by a polynomial of b. Finally, by the standard arguments, [15, Theorem 1.6], based on a polynomial reduction between any NP-complete problems, we obtain a polynomial kernel for MWBS.

By pipelining the crucial step of the kernelization algorithm with the branchwidth algorithm, we obtain a parameterized subexponential algorithm for MWBS of running time $2^{O(\sqrt{b})} \cdot n^{O(1)}$. Since $b \leq n$, this also implies an algorithm of running time $2^{O(\sqrt{n})}$. We remark that our algorithms are asymptotically optimal up to the *Exponential Time Hypothesis* (ETH) [16, 17]. The NP-hardness result of MBS (and hence of MWBS) given in [8] exploits a reduction from Planar-3SAT. The number of non-bimodal vertices in the resulting instance of MBS is linear in the size of the Planar-3SAT instance. Using the standard techniques for computational lower bounds for problems on planar graphs [10], we obtain that the existence of an $2^{o(\sqrt{b})} \cdot n^{O(1)}$-time algorithm for MBWS would contradict ETH.

- *Approximability.* We provide an Efficient Polynomial-Time Approximation Scheme (EPTAS) for MWBS, based on Baker's (or shifting) technique [4]. More precisely, using our algorithm for graphs of bounded branchwidth, we give an $(1+\epsilon)$-approximation algorithm that runs in $2^{O(1/\epsilon)} \cdot n^{O(1)}$ time.

Future research

A natural future research direction is to extend our study to a generalization of bimodality, called *k-modality*. Given a positive even integer k, a plane digraph is *k-modal* if the edges at each vertex can be grouped into at most k sets of consecutive edges with the same orientation [20]. For example, it is known that 4-modality is necessary for planar L-drawings [9].

References

1 Patrizio Angelini, Steven Chaplick, Sabine Cornelsen, and Giordano Da Lozzo. Planar L-drawings of bimodal graphs. *J. Graph Algorithms Appl.*, 26(3):307–334, 2022.

2 Patrizio Angelini, Steven Chaplick, Sabine Cornelsen, and Giordano Da Lozzo. On upward-planar l-drawings of graphs. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 10:1–10:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

3 Patrizio Angelini, Giordano Da Lozzo, Marco Di Bartolomeo, Valentino Di Donato, Maurizio Patrignani, Vincenzo Roselli, and Ioannis G. Tollis. Algorithms and bounds for L-drawings of directed graphs. *Int. J. Found. Comput. Sci.*, 29(4):461–480, 2018.

4 Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. Assoc. Comput. Mach.*, 41(1):153–180, 1994.

5 Paola Bertolazzi, Giuseppe Di Battista, and Walter Didimo. Quasi-upward planarity. *Algorithmica*, 32(3):474–506, 2002.

6 Carla Binucci, Emilio Di Giacomo, Giuseppe Liotta, and Alessandra Tappini. Quasi-upward planar drawings with minimum curve complexity. In Helen C. Purchase and Ignaz Rutter, editors, *Graph Drawing and Network Visualization – 29th International Symposium, GD 2021, Tübingen, Germany, September 14-17, 2021, Revised Selected Papers*, volume 12868 of *Lecture Notes in Computer Science*, pages 195–209. Springer, 2021.

7 Carla Binucci and Walter Didimo. Computing quasi-upward planar drawings of mixed graphs. *Comput. J.*, 59(1):133–150, 2016.

8 Carla Binucci, Walter Didimo, and Francesco Giordano. Maximum upward planar subgraphs of embedded planar digraphs. *Comput. Geom.*, 41(3):230–246, 2008.

9 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. In Fabrizio Frati and Kwan-Liu Ma, editors, *Graph Drawing and Network Visualization – 25th International Symposium, GD 2017, Boston, MA, USA, September 25-27, 2017, Revised Selected Papers*, volume 10692 of *Lecture Notes in Computer Science*, pages 465–478. Springer, 2017.

10 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

11 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice-Hall, 1999.

12 Giuseppe Di Battista and Enrico Nardelli. Hierarchies and planarity theory. *IEEE Trans. Syst. Man Cybern.*, 18(6):1035–1046, 1988.

13 Walter Didimo. Upward graph drawing. In *Encyclopedia of Algorithms*, pages 2308–2312. 2016.

14 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010.

15 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization. Theory of parameterized preprocessing.* Cambridge University Press, Cambridge, 2019.

16 Russell Impagliazzo and Ramamohan Paturi. Complexity of k-sat. In *Proceedings of the 14th Annual IEEE Conference on Computational Complexity, Atlanta, Georgia, USA, May 4-6, 1999*, pages 237–240. IEEE Computer Society, 1999.

17 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

18 Michael Jünger, Sebastian Leipert, and Petra Mutzel. Level planarity testing in linear time. In Sue Whitesides, editor, *Graph Drawing, 6th International Symposium, GD'98, Montréal, Canada, August 1998, Proceedings*, volume 1547 of *Lecture Notes in Computer Science*, pages 224–237. Springer, 1998.

19 Neil Robertson and Paul D. Seymour. Graph minors. X. obstructions to tree-decomposition. *J. Comb. Theory, Ser. B*, 52(2):153–190, 1991.

20 Juan José Besa Vial, Giordano Da Lozzo, and Michael T. Goodrich. Computing k-modal embeddings of planar digraphs. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 19:1–19:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

5.4 Parameterised Algorithms for Sequential Crossing Number

Eduard Eiben (Royal Holloway, University of London, GB), Thekla Hamm (Utrecht University, NL), Petr Hliněný (Masaryk University – Brno, CZ), Martin Nöllenburg (TU Wien, AT), Ignaz Rutter (Universität Passau, DE), and Manuel Sorge (TU Wien, AT)

A driving motivation of the field of graph drawing is the visualisation of information that can be captured by graphs. Often such information can evolve over time and hence the treatment of temporal aspects is an important challenge in graph drawing. In this context it makes sense to require some "stability" of the drawing of a graph that changes over time in the sense that parts of the graph continue to occur in consecutive time steps are not redrawn in a different way. Motivated by this we define the following crossing-minimal drawing variant. Below $\mathrm{cr}(\mathcal{G})$ for a drawing $\mathcal{G}$ of a graph denotes the number of edge crossings in the drawing.

Sequential Crossing Number (SCN):
Problem: A sequence $G_1, G_2, \ldots, G_t$ of (in general non-disjoint) graphs, an integer k.
Question: Is there a drawing $\mathcal{G}_i$ of each G_i such that for each $i \in [t-1]$ we have

$$\mathcal{G}_i|_{G_i \cap G_{i+1}} = \mathcal{G}_{i+1}|_{G_i \cap G_{i+1}} \text{ and } \sum_{i \in [t]} \mathrm{cr}(\mathcal{G}_i) \leq k?$$

Relations to other problems and implied hardness results

SCN is obviously more general than the classical crossing number problem, which corresponds to the setting in which $t = 1$, and it is also more general than the partially predrawn crossing number problem [5] which can be captured using $t = 2$, having as G_1 the rigidised (also adding many parallel edges) prescribed partial drawing of the graph and as G_2 the graph to which the partial predrawing should be extended.

Moreover, in the special case that for all $i, j \in [t]$, $G_i \cap G_j = \bigcap_{\ell \in [t]} G_i$, i.e. the pairwise intersection of all given graphs is the same, then we speak of *sunflower instances*, and refer to $\bigcap_{\ell \in [t]} G_i$ as the *core* (in line with terminology used for Simultaneous Embedding with Fixed Edges (SEFE) [2]). On sunflower instances, SEFE and SCN with $k = 0$ are easily seen to be completely equivalent.

Indeed, there is even a simple reduction from SEFE on general (not necessarily sunflower) instances to SCN with $k = 0$ by considering, for an input $G_1, \ldots, G_t$ of SEFE, the instance of SCN (on sunflower instances) given by $G_\ell \cup \bigcup_{i,j \in [t]} G_i \cap G_j$, for $\ell \in [t]$, and $k = 0$.

From the known hardness of SEFE on sunflower instances with $t = 3$ and the core being a star[1], we obtain the following.

▶ **Theorem 1.** *SCN is* NP*-hard, even on sunflower instances with the core being a star, $t = 3$ and $k = 0$.*

A generalisation of SEFE to allow for drawings with crossings in which the total number of crossings over any single graph is considered as the decision value has been proposed as Simultaneous Crossing Number [3], and can be reduced to SCN (with general k) in the same way. Of course, since both problems are NP-complete, there is also a polynomial reduction from SCN to the simultaneous crossing number problem. However, a difference between both reductions is that, in the direction from Simultaneous Crossing Number to SCN, we never increase the structural complexity of the union of all instance graphs, whereas this is necessary in the direction from SCN to Simultaneous Crossing Number to allow for 'reappearing' parts of graphs to be drawn differently.

SCN with $k = 0$ also generalises planarity of streamed graphs [4]. A *streamed graph* with *time window* $w \in \mathbb{N}$ and backbone is given by a vertex set V, a set of edges B on V called a *backbone*, and an ordered non-repeating sequence of edges $e_1, \ldots, e_m$ on V which are not in B. By definition, this streamed graph is planar if and only if the following is a yes-instance for SCN with $k = 0$: For $t = m$ and for $i \in [t]$, $G_i := (V, B \cup \{e_j \mid i \leq j < i + w\})$.

From the hardness result for testing planarity of streamed graphs with time window 5 and empty backbones [4] we can infer the following for SCN.

▶ **Theorem 2.** *SCN is* NP*-hard, even on instances with $|E(G_i)| \leq 5$ for all $i \in [t]$ and $k = 0$.*

Parameterised Algorithms for Sequential Crossing Number

In view of the above hardness results, our main target during the seminar was to formulate parameterised algorithms for SCN. Notably, we believe to be able to give the following results.

1. SCN parameterised by $k + \max_{i \in [t-1]} |V(G_i) \cap V(G_{i+1})|$ is in FPT.
2. SCN parameterised by k is in XP on inputs in which for all $i \in [t-1]$, $G_i \cap G_{i+1}$ is 3-connected. (This is subsumed by the next FPT result but also used in its proof with $k = 0$.)

3. SCN parameterised by k is in FPT on inputs in which for all $i \in [t-1]$, $G_i \cap G_{i+1}$ is 3-connected.
4. SCN parameterised by $k + t + r$ is in FPT, where r is an upper bound on the maximum number of bridges of a P-node in the SPQR-tree and the maximum degree of a vertex in the block cut tree of the intersection of two consecutive input graphs, on instances in which for all $i \in [t-1]$, $G_i \cap G_{i+1}$ is connected.
   - Notice that for 3-connected consecutive intersections, $r = 0$ and the condition is obviously satisfied. The only reason that this result does not subsume the FPT-result for 3-connected consecutive intersections is the additional parameterisation on t.
   - The hardness from Theorem 1 implies that we cannot hope to drop r – the maximum degree of a vertex in the block cut tree of the intersection of two consecutive input graphs, from the problem parameter.

In terms of techniques, the first two results are achieved by dynamic programming and branching algorithms, respectively, the former of which we briefly sketch:

▶ **Theorem 3.** *SCN parameterised by* $k + \max_{i \in [t-1]} |V(G_i) \cap V(G_{i+1})|$ *is in* FPT.

Idea. We carry out dynamic programming on t, where the dynamic programming table is indexed by pairs of a drawing of $G_{i-1} \cap G_i$ and a drawing of $G_i \cap G_{i+1}$ with at most k crossings in total. The number of such pairs of drawings is easily seen to be bounded in the considered parameter. Each table entry stores the minimum total number of crossings so far (using the state from the previous dynamic programming step) under the condition that the drawing of G_i coincides with the index-prescribed drawings of $G_{i-1} \cap G_i$ and $G_i \cap G_{i+1}$. To compute an entry for a table index we use the FPT-algorithm for partially predrawn crossing number [5]. ◀

The second two results are far more involved: Similarly to the fixed-parameter algorithm we use a bicriteria approach to reduce the treewidth of the input graphs to a function of the considered parameters – and, more importantly, the treewidth of a derived auxiliary graph that arises from the disjoint union of all G_i by identifying vertices in the intersections (both edges and vertices) of consecutive G_i with one another. For Result 3 this needs to be preceded by a careful reduction of t to a function of k using Result 2. Once we are in a situation of bounded treewidth, we want to invoke Courcelle's theorem on a bounded-length MSO-formula that encodes the fact that the considered auxiliary graph has a drawing with at most k-crossings. However, previously no such formula was known. To overcome this difficulty, we formulate a result of independent interest:

- We give an $f(k)$-length MSO-formula for expressing the existence of a drawing of a connected graph G with at most k crossings on a graph derived in polynomial time from G in a way that crossings and rotations around cut vertices of P-nodes and high-degree cut vertices are given by the interpretation of the free variables.

The crucial advantage over the known formulations for expressing the existence of a drawing of a connected graph G with at most k crossings via excluding Kuratowski minors after identifying crossings (which is also used in the known FPT-algorithm for Crossing Number) is that the known formulations give us no direct knowledge of the drawing. Let us highlight the fact that the graph derived from G has treewidth bounded in $\text{tw}(G) + r$, which is where r comes in as a parameter in Result 4.

References

1 P. Angelini, G. Da Lozzo and D. Neuwirth, *Advancements on SEFE and Partitioned Book Embedding problems*, Theoretical Computer Science, 2015, pp. 71-89, doi: 10.1016/j.tcs.2014.11.016.

2 T. Bläsius, S. G. Kobourov and I. Rutter, *Simultaneous Embedding of Planar Graphs*, Handbook on Graph Drawing and Visualization, 2013, pp.349-381.

3 M. Chimani, M. Jünger and M. Schulz, *Crossing Minimization meets Simultaneous Drawing*, 2008 IEEE Pacific Visualization Symposium, 2008, pp. 33-40, doi: 10.1109/PACIFICVIS.2008.4475456.

4 G. Da Lozzo and I. Rutter, *Planarity of streamed graphs*, Theoretical Computer Science, 2019, pp. 1-21, doi: 10.1016/j.tcs.2019.09.029.

5 T. Hamm and P. Hliněný, *Parameterised Partially-Predrawn Crossing Number*, 38th International Symposium on Computational Geometry (SoCG 2022), 2022, doi: 10.4230/LIPIcs.SoCG.2022.46.

5.5 Upward and Rectilinear Planarity Testing and *st*-Planar Edge Completion

Bart M. P. Jansen (TU Eindhoven, NL), Liana Khazaliya (TU Wien, AT), Philipp Kindermann (Universität Trier, DE), Giuseppe Liotta (University of Perugia, IT), Fabrizio Montecchiani (University of Perugia, IT), Kirill Simonov (Hasso Plattner Institute, University of Potsdam, DE)

The first problems considered by this working group are Upward and Rectilinear Planarity Testing. Given a directed acyclic graph G, upward planarity testing asks whether G admits a crossing-free drawing where all edges are monotonically increasing in a common direction, which is conventionally called the upward direction; see Fig.3. For an undirected graph G, rectilinear planarity testing asks whether G admits a crossing-free drawing such that each edge is either a horizontal or a vertical segment; see Fig.3. Both upward planarity and rectilinear planarity testing are classical and extensively investigated topics in graph drawing (see, e.g. [1, 19, 15, 18]). It is known that they are both NP-complete in the so-called *variable embedding setting*, that is when the testing algorithm must verify whether the input graph has a combinatorial structure of its faces which allows the balancing of angles described above. Again unsurprisingly, both proofs of NP-completeness follow the same logic based on a reduction from a common flow problem on planar graphs [20]. These NP-completeness results have motivated a flourishing literature describing both polynomial-time solutions for special classes of graphs and parameterized solutions for general graphs. For example, polynomial-time solutions are known for both problems when the input graph has treewidth at most two [10, 12, 13, 6, 11]; also, rectilinear planarity testing can be solved in linear time if the maximum degree of the input graph is at most three [14], and upward planarity testing can be solved in linear time if the digraph has only one source vertex [2, 4, 21]. Concerning parameterized solutions, upward planarity testing is fixed-parameter tractable when parameterized by the number of triconnected components [5], by the treedepth [7], and by the number of sources [7].

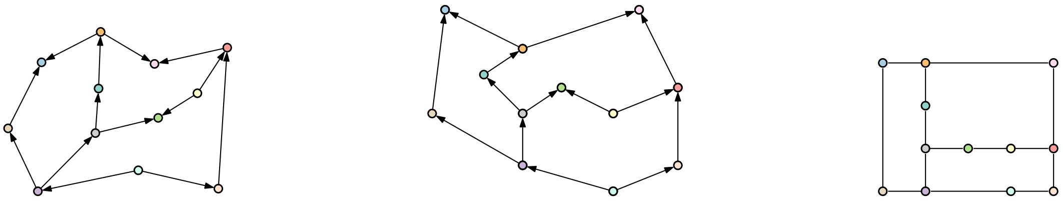

Figure 3 A directed graph $\overrightarrow{G}$, an upward planar drawing of $\overrightarrow{G}$, and a rectilinear planar drawing of its underlying undirected graph G.

The research in this paper is motivated by the fact that both upward planarity and rectilinear planarity testing are known to lie in XP when parameterized by treewidth [7, 11]. Determining whether these two parameterized problems are in FPT are mentioned as open problems in both [7, 11]. The main contribution of this paper is as follows.

▶ **Theorem 1.** *Upward planarity testing and rectilinear planarity testing parameterized by treewidth are both W[1]-hard. Moreover, assuming the Exponential Time Hypothesis, neither problem can be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f, where k is the treewidth of the input graph.*

Fig.1 implies that, under the standard hypothesis FPT $\neq$ W[1] in parameterized complexity, there exists no fixed-parameter tractable algorithm for either problem parameterized by treewidth, hence answering the above mentioned open problems. To obtain our results we analyze the auxiliary flow problem used as a common starting point in the NP-completeness proof of both planarity problems. It closely resembles the All-or-Nothing Flow problem (AoNF), which asks for an st-flow of prescribed value in an edge-capacitated flow network such that each edge is either used fully, or not at all. The AoNF problem parameterized by treewidth was recently shown to be W[1]-hard (in fact, even XNLP-complete) on general graphs by Bodlaender et al. [3]. By a significant adaptation of their construction, we can prove that AoNF parameterized by treewidth remains W[1]-hard on *planar* graphs. By revisiting the chain of reductions to the planarity testing problems, passing through the Circulating Orientation problem in between, we show they can be carried out without blowing up the treewidth of the graph and thereby obtain Fig.1.

Last but not least, we also considered the related st-Planar Edge Completion Problem. Let $G = (V, E)$ be a digraph. A vertex of G with no incoming edges is a *source* of G, while a vertex without outgoing edges is a *sink* of G. A digraph G is an *st-planar graph* if it admits a planar embedding such that: (1) it contains no directed cycle; (2) it contains a single source vertex s and a single sink vertex t; (3) s and t both belong to the external face of the planar embedding.

Upward planarity, as was also mentioned above, is a rather natural and well-studied notion of planarity for directed graphs (see, e.g., [7, 8, 1, 9, 20, 16]). In particular, a planar digraph is *upward* if it admits a planar drawing where all edges are oriented upward. A well-known result in graph drawing states that a digraph G is upward if and only if G is a subgraph of an st-planar graph [1, 9][2]. However, since testing for upward planarity is an NP-complete problem already for biconnected graphs [20], determining whether a biconnected

[2] From the proof in Lemma 4.1 of [9], one can in fact observe that a digraph is upward planar if and only if it is a subgraph of an st-planar graph defined over the same set of vertices.

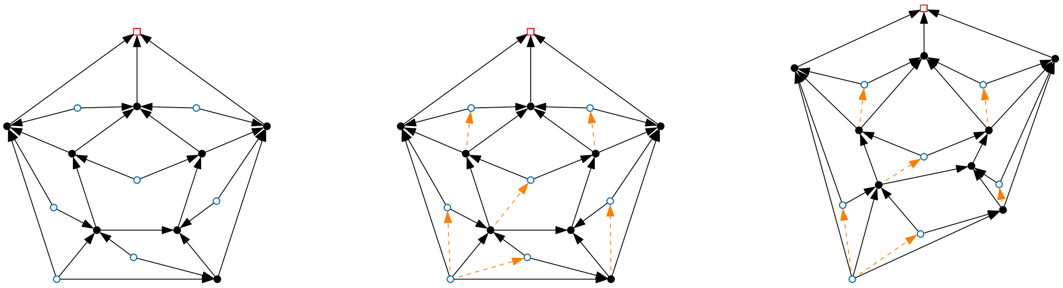

Figure 4 (a) A digraph G with $2k + 1 = 7$ sources and 1 sink; G has a unique planar embedding up to the choice of the external face; (b) A completion of G to an st-planar graph obtained by adding $2k = 6$ edges; (c) An upward planar drawing of the completion of G.

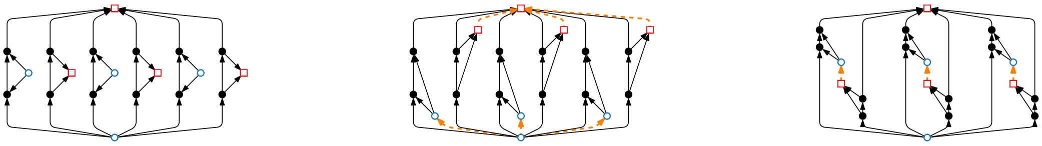

Figure 5 (a) A biconnected digraph G with 4 sources and 4 sinks; (b) With the given embedding, 6 edges have to be added to complete G to an st-planar graph; (c) With a different embedding, adding 3 edges is sufficient.

graph is a subgraph of an st-planar graph is also computationally challenging. On the other hand, checking whether a digraph is st-planar can be done efficiently in polynomial time. This observation motivates for the investigation of the following problem.

st-Planar Edge Completion (st-PEC)
Input: A biconnected digraph G
Parameter: $k \in \mathbb{N}$
Question: Is it possible to add at most k edges to G such that the resulting graph is an st-planar graph?

We came up with a fixed-parameter tractable algorithm for the st-Planar Edge Completion problem. To help understanding the combinatorial and algorithmic challenges behind the problem, we make the observation that the parameter k provides an upper bound on the number of sources and sinks in the input digraph G. Since an edge can remove the presence of at most one source and one sink, if the total number of sources and sinks in G exceeds $2k + 2$, we can promptly reject the instance. Conversely, a positive answer to st-Planar Edge Completion implies that G is upward planar. In this respect, it is worth mentioning that Chaplick et al. [7] have previously demonstrated that testing a digraph for upward planarity is fixed-parameter tractable when parameterized by the number of its sources. However, for every $k \geq 1$, there are upward planar digraphs with at most $2k + 1$ sources that cannot be augmented to an st-planar graph by adding k edges; refer to Fig.4 for an illustration. Furthermore, while an upward planarity test halts upon finding an upward planar embedding, not all upward planar embeddings of the same digraph can lead to an st-planar graph after the addition of k edges. Fig.5 demonstrates an upward planar digraph along with two of its upward planar embeddings: the embedding in Fig.5 requires 6 edges to be augmented into an st-planar digraph, whereas the embedding in Fig.5 can be augmented with 3 edges.

In order to overcome the above technical challenges, our result is based on a structural decomposition of the digraph into its triconnected components using SPQR-trees (similarly as done in [7]), as well as on novel insights regarding the combinatorial properties of upward planar digraphs.

References

1 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, Ioannis G. Tollis. Graph Drawing: Algorithms for the Visualization of Graphs. Prentice-Hall, 1999, 0-13-301615-3, https://dblp.org/rec/bib/books/ph/BattistaETT99.

2 Paola Bertolazzi, Giuseppe Di Battista, Carlo Mannino, Roberto Tamassia. Optimal Upward Planarity Testing of Single-Source Digraphs. SIAM J. Comput. https://dblp.org/rec/journals/siamcomp/BertolazziBMT98.

3 Hans L. Bodlaender, Gunther Cornelissen, Marieke van der Wegen. Problems hard for treewidth but easy for stable gonality. Graph-Theoretic Concepts in Computer Science – 48th International Workshop, WG 2022, Tübingen, Germany, June 22-24, 2022, Revised Selected Papers, Lecture Notes in Computer Science, 13453, 84–97, Springer, 2022.

4 Guido Brückner, Markus Himmel, Ignaz Rutter. An SPQR-Tree-Like Embedding Representation for Upward Planarity. Graph Drawing and Network Visualization – 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings. https://dblp.org/rec/conf/gd/BrucknerHR19.bib.

5 Hubert Y. Chan, A Parameterized Algorithm for Upward Planarity Testing, Algorithms – ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings, Lecture Notes in Computer Science,10.1007/978-3-540-30140-0_16.

6 Fabrizio Frati, Planar rectilinear drawings of outerplanar graphs in linear time, Comput. Geom., 10.1016/j.comgeo.2021.101854.

7 Steven Chaplick and Emilio Di Giacomo and Fabrizio Frati and Robert Ganian and Chrysanthi N. Raftopoulou and Kirill Simonov, Parameterized Algorithms for Upward Planarity, 38th International Symposium on Computational Geometry, SoCG 2022, June 7-10, 2022, Berlin, Germany, LIPIcs, 224, 26:1–26:16, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 10.4230/LIPIcs.SoCG.2022.26.

8 Steven Chaplick and Emilio Di Giacomo and Fabrizio Frati and Robert Ganian and Chrysanthi N. Raftopoulou and Kirill Simonov, Testing Upward Planarity of Partial 2-Trees, GD, Lecture Notes in Computer Science, 13764, 175–187, Springer, 2022.

9 Giuseppe Di Battista and Roberto Tamassia, Algorithms for Plane Representations of Acyclic Digraphs, Theor. Comput. Sci., 61, 175–198, 1988.

10 Giuseppe Di Battista and Giuseppe Liotta and Francesco Vargiu, Spirality and Optimal Orthogonal Drawings, SIAM J. Comput., 27, 6, 1764–1811, 1998.

11 Emilio Di Giacomo and Giuseppe Liotta and Fabrizio Montecchiani, Orthogonal planarity testing of bounded treewidth graphs, J. Comput. Syst. Sci., 125, 129–148, 2022, 10.1016/j.jcss.2021.11.004.

12 Walter Didimo and Francesco Giordano and Giuseppe Liotta, Upward Spirality and Upward Planarity Testing, SIAM J. Discret. Math., 23, 4, 1842–1899, 2009, 10.1137/070696854.

13 Walter Didimo and Michael Kaufmann and Giuseppe Liotta and Giacomo Ortali, Rectilinear Planarity of Partial 2-Trees, GD, Lecture Notes in Computer Science, 13764, 157–172, Springer, 2022.

14 Walter Didimo and Giuseppe Liotta and Giacomo Ortali and Maurizio Patrignani, Optimal Orthogonal Drawings of Planar 3-Graphs in Linear Time, Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020, 806–825, SIAM, 2020, 10.1137/1.9781611975994.49.

15 Takao Nishizeki and Md. Saidur Rahman, Planar Graph Drawing, Lecture Notes Series on Computing, 12, World Scientific, 2004, 10.1142/5648, 981-256-033-5, Mon, 03 Apr 2023 16:01:56 +0200, https://dblp.org/rec/books/ws/NishizekiR04.bib.

16 William T. Trotter and John I. Moore Jr., The dimension of planar posets, J. Comb. Theory, Ser. B, 22, 1, 54–67, 1977.

17 Roberto Tamassia, Handbook on Graph Drawing and Visualization, 1–42, 2013, https://cs.brown.edu/people/rtamassi/gdhandbook/chapters/planarity.pdf.

18 Roberto Tamassia, Handbook on Graph Drawing and Visualization, 1–42, 2013, https://cs.brown.edu/people/rtamassi/gdhandbook/chapters/planarity.pdf.

19 Michael Kaufmann and Dorothea Wagner, Drawing Graphs, Methods and Models (the book grow out of a Dagstuhl Seminar, April 1999), Lecture Notes in Computer Science, 2025, Springer, 2001, 10.1007/3-540-44969-8, https://dblp.org/rec/conf/dagstuhl/1999dg.bib.

20 Ashim Garg and Roberto Tamassia, On the Computational Complexity of Upward and Rectilinear Planarity Testing, SIAM J. Comput., 31, 2, 601–625, 2001.

21 Michael D. Hutton and Anna Lubiw, Upward Planning of Single-Source Acyclic Digraphs, SIAM J. Comput., 25, 2, 291–311, 1996.

Participants

- Akanksha Agrawal
Indian Institute of Techology Madras, IN
- Sergio Cabello
University of Ljubljana, SI
- Giordano Da Lozzo
University of Rome III, IT
- Walter Didimo
University of Perugia, IT
- Eduard Eiben
Royal Holloway, University of London, GB
- Fedor V. Fomin
University of Bergen, NO
- Robert Ganian
TU Wien, AT
- Petr A. Golovach
University of Bergen, NO
- Siddharth Gupta
University of Warwick – Coventry, GB
- Thekla Hamm
Utrecht University, NL
- Petr Hlinený
Masaryk University – Brno, CZ
- Tanmay Inamdar
University of Bergen, NO
- Bart Jansen
TU Eindhoven, NL
- Michael Kaufmann
Universität Tübingen, DE
- Liana Khazaliya
TU Wien, AT
- Philipp Kindermann
Universität Trier, DE
- Stephen G. Kobourov
University of Arizona – Tucson, US
- Giuseppe Liotta
University of Perugia, IT
- Bojan Mohar
Simon Fraser University – Burnaby, CA
- Fabrizio Montecchiani
University of Perugia, IT
- Martin Nöllenburg
TU Wien, AT
- Sebastian Ordyniak
University of Leeds, GB
- Ignaz Rutter
Universität Passau, DE
- Saket Saurabh
The Institute of Mathematical Sciences – Chennai, IN
- Raimund Seidel
Universität des Saarlandes – Saarbrücken, DE
- Roohani Sharma
MPI für Informatik – Saarbrücken, DE
- Marie Diana Sieper
Universität Würzburg, DE
- Kirill Simonov
Hasso-Plattner-Institut, Universität Potsdam, DE
- Manuel Sorge
TU Wien, AT
- Yushi Uno
Osaka Metropolitan University, JP
- Alexander Wolff
Universität Würzburg, DE
- Meirav Zehavi
Ben Gurion University – Beer Sheva, IL