

Parameterized Complexity in Graph Drawing

Edited by

Robert Ganian¹, Fabrizio Montecchiani², Martin Nöllenburg³, and Meirav Zehavi⁴

- 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, IL, zehavimeirav@gmail.com

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 21293 “Parameterized Complexity in Graph Drawing”. The seminar was held mostly in-person from July 18 to July 23, 2021. It brought together 28 researchers from the Graph Drawing and the Parameterized Complexity research communities with the aim to discuss and explore open research questions 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 July 18–23, 2021 – <http://www.dagstuhl.de/21293>

2012 ACM Subject Classification Theory of computation → Computational geometry; Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases exact computation, graph algorithms, graph drawing, parameterized complexity

Digital Object Identifier 10.4230/DagRep.11.6.82

Edited in cooperation with Jules Wulms

1 Executive Summary

Robert Ganian (TU Wien, AT)

Fabrizio Montecchiani (University of Perugia, IT)

Martin Nöllenburg (TU Wien, AT)

Meirav Zehavi (Ben-Gurion University, IL)

License © Creative Commons BY 4.0 International license
© Robert Ganian, Fabrizio Montecchiani, Martin Nöllenburg, and Meirav Zehavi

Graph Drawing. Graph-based models are pervasive in many fields of science and technology. Very often scientists and users analyze these models and communicate their findings by means of graphical representations. This motivated the birth and evolution of *graph drawing*, a self-standing discipline that has evolved tremendously over the past 50 years. Today graph drawing is a mature area of computer science [5, 13, 17, 18] with its own annual conference, the International Symposium on Graph Drawing and Network Visualization (GD)¹. The focus of the research area today is on combinatorial and algorithmic aspects of drawing graphs as well as on the design of network visualization systems and interfaces. Graph

¹ see www.graphdrawing.org

drawing is motivated by applications where it is crucial to visually analyze and interact with relational datasets. Examples of such application areas include data science, social sciences, web computing, information systems, biology, geography, business intelligence, information security and software engineering.

Roughly speaking, graph drawing deals with the construction and analysis of geometric representations of graphs and networks subject to specific layout conventions, such as different notions of planarity or more general crossing constraints, grid layouts, orthogonal drawings etc. Many classic graph drawing problems are NP-hard and thus a variety of theoretical and practical algorithmic techniques for dealing with hard problems are required in graph drawing.

Parameterized Complexity. Numerous computational problems of wide interest are known to be NP-hard in general. Yet, it is often possible to utilize the structure implicitly underlying many real-world instances to find exact solutions efficiently. There is long-standing systematic research of tractability results for various problems on specific classes of instances, and research in this direction constitutes one of the fundamental areas of computer science. However, in many real-world situations it is not possible to define a clear-cut class of instances that we wish to solve; instead of being black and white (belonging to a specific class or not), instances often come in various shades of grey (having certain degrees of internal structure).

The relatively young *parameterized complexity* paradigm [6, 4, 8, 16] offers the perfect tools to deal with this situation. In the parameterized setting, we associate each instance with a numerical *parameter*, which captures how “structured” the instance is. This then allows the development of algorithms whose performance strongly depends on the parameter – instead of the classical setting, where we often associate tractability with polynomial running times and intractability with superpolynomial ones, parameterized algorithms naturally “scale” with the amount of structure contained in the instance. The central notion of tractability in the parameterized setting is *fixed-parameter tractable* (FPT in short), which means that the given problem can be solved by an algorithm with runtime of the form $f(k) \cdot n^{\mathcal{O}(1)}$ (where f is an arbitrary computable function, k is the value of the parameter, and n is the input size). Aside from fixed-parameter tractability, the parameterized complexity landscape consists of a variety of companion notions such as *XP-tractability*, *kernelization* and *W-hardness*.

Parameterized Complexity in Graph Drawing. Research at the intersection of graph drawing and parameterized complexity (and parameterized algorithms in particular) is in its infancy. Most of the early efforts have been directed at variants of the classic Crossing Minimization problem, introduced by Turán in 1940 [19], parameterized by the number of crossings. Here, the objective is to draw a given graph in the plane so as to induce minimum number of crossings. Already in 2001, it was shown to be FPT [9]. A few subsequent works followed [14, 11], including the best paper of GD 2019 [12], but also concerning restricted layouts such as two-layered embeddings [7] and two-sided circular graph layouts [15]. On a related note, given a graph drawn in the plane, some preliminary works considered the detection of a subgraph having a particular structure with minimum number of crossings [1, 10]. Recently, parameterized analysis of specific embeddings such as book embeddings [3, 2], was also brought into life. Overall, the intersection of graph drawing and parameterized complexity still remains mostly unexplored, yet we see many interesting challenges and opportunities for taking a parameterized perspective on graph drawing problems and investigating the applicability of advanced parameterized techniques.

Seminar Goals

The main goal of the seminar was to chart new paths towards research combining the latest findings and techniques in parameterized complexity and graph drawing. In particular, the seminar focused on several prominent topics in graph drawing as well as state-of-the-art tools in parameterized complexity. The discussions addressed both concrete open problems as well as general directions for future research. An integral part of these discussions was the identification and formulation of major challenges as well as novel parameterizations of graph drawing problems relevant to parameterized analysis. The discussions also addressed the applicability of classic as well as cutting-edge tools in parameterized complexity to graph drawing.

In view of the above, it is safe to say that the selection of suitable problems to target was of great importance for the success of the seminar. Our main aim was to offer the participants the opportunity to propose problems to work on, and so the final selection of problems targeted by working groups was carried out during the seminar itself. That being said, we have also prepared a list of candidate problems that we believe would be prime candidates for further investigation through the lens of parameterized complexity.

Seminar Program

1. On the first day of the seminar we enjoyed short introductions of all participants, and four invited overview lectures on different research domains within Graph Drawing. The topics and speakers were chosen as to create a joint understanding of the state of the art of problems in Graph Drawing suitable for parameterized analysis. Thekla Hamm presented the topic of graph drawing extension problems, Petr Hliněný presented the topic of planar insertion problems, Michael Kaufmann presented the topic of graph drawing beyond planarity and parameterized complexity, and Ignaz Rutter presented the topic of constrained embedding problems. More information on each lecture can be found in Section 3. Overall, this day prepared the ground for the open problem session on the second day.
2. The open problem session took place in the morning of the second day of the seminar. In this session, we collected a list of open research problems that were contributed by the seminar participants. In a preference voting we determined the five topics that raised the most interest among the participants and formed small working groups around them. Each group contained experts in both Graph Drawing and Parameterized Complexity. During the following days the groups worked by themselves, except for a few plenary reporting sessions, formalizing and solving their respective challenges. Below is a list of the working group topics; more detailed group reports are found in Section 5.
 - a. **Upward/level planarity:** This group studied two previously established restrictions of drawing planar graphs: vertices are either assigned a “horizontal level” that they must be placed on, or there are directed arcs and the drawing must have all edges facing upwards. The group aimed at the development of new parameterized algorithms for both of these NP-hard problems.
 - b. **Two-page embeddings of upward planar graphs:** The group studied the complexity of recognizing whether *st*-planar graphs admit an upward two-page book embedding.
 - c. **Orthogonal drawings:** The group focused on the COMPACT problem (computing a minimum-area drawing for an orthogonal graph), parameterized primarily by the number of kitty corners, that is, pairs of reflex vertices that point to each other.

- d. **Almost Separated Fixed Order Stack Layouts:** In a fixed order stack layout the vertices of a graph are given with a fixed order and one has to assign the edges to pages so that no two edges on any page cross. This group studied a variant of this well-known NP-complete problem, where the graph is bipartite and the vertices form k consecutive blocks from either part.
 - e. **Graph product structure theorem:** The group considered strong products of graphs that yield supergraphs of k -planar graphs, i.e. of graphs that admit a drawing in the plane in which each edge is crossed at most k times. The objective is to exploit these products to derive new upper bounds on the queue number of k -planar graphs.
 - f. **Decision trees:** Decision Trees are well known tools used to describe, classify, and generalize data. Besides their simplicity, decision trees are particularly attractive for providing interpretable models of the underlying data. The group studied the complexity of learning decision trees of minimum size under several different parameterizations.
3. After the open problem session, Robert Ganian gave a tutorial in the second day of the seminar that showcased how some of the tools in Parameterized Complexity can be applied to difficult problems, with a special focus on problems that are relevant to graph drawing. The tutorial was prepared in a way so as to make it accessible to the graph drawing community, acting as catalysis for progress on the five selected topics.
 4. In the rest of the second day and the other days of the seminar, we had a flexible working schedule with a short plenary session every morning to accommodate group reports and impromptu presentations by participants.

Future Plans

The seminar was designed to foster new research collaborations between researchers in the graph drawing and parameterized complexity communities, whose paths rarely cross in the traditional conferences. These collaborations are very likely to result in new breakthroughs and results, and we expect that the seminar will lead to tangible progress in our understanding of problems of interest. In this sense, the primary outcome from the seminar will be research papers published at the core conferences and journals for the graph drawing and parameterized complexity communities, such as:

- 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 Theoretical Aspects of Computer Science (**STACS**).

In the mid- and long-term horizon, the seminar will also help build a bridge between the two communities and identify other interesting graph drawing problems which would benefit from a rigorous investigation using tools from parameterized complexity. It 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. Last but not least, the seminar will raise the awareness for the typical research problems and the latest techniques in each others community and thus enrich the knowledge and toolbox of individual participants.

Dagstuhl seminar in 2022/2023 on Graph Drawing in Parameterized Complexity. This Dagstuhl seminar has revealed, for the first time in a systematic way, the astounding wealth of problems in Graph Drawing that are naturally multivariate and hence suitable

for parameterized analysis; thus a follow-up Dagstuhl seminar will be proposed to further discuss and deepen our understanding of this topic whose full potential is yet to be unlocked, once again bringing together researchers in Graph Drawing and Parameterized Complexity.

Evaluation

According to the Dagstuhl survey conducted after the seminar, as well as informal feedback to the organizers, the seminar was highly appreciated. Particularly the small group size, group composition, and the seminar structure focusing on hands-on working groups was very well received. The seminar's goals to identify new research directions and initiate collaborations at the intersection of the two different fields of Graph Drawing and Parameterized Complexity was very successful (also in comparison to other Dagstuhl seminars). Indeed, the participants rated the seminar highly for the mixture of these two fields and its productive interdisciplinary atmosphere, yielding new research perspectives, which have also resulted in new collaborations, joint projects and publications. 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.

The seminar had more participants from the Graph Drawing community than from the Parameterized Complexity community due to critical uncertainties caused by the COVID-19 pandemic. We hope that the current trend of improvement in the situation will help in composing a more balanced list of participants in future seminars on this topic.

Acknowledgments

Schloss Dagstuhl was the perfect place for hosting a seminar like this. The unique scientific atmosphere and the historic building provided not only all the room we needed for our program and the working groups, but also plenty of opportunities for continued discussions and socializing outside the official program, especially in these difficult times during the COVID-19 pandemic with all participants being eager to meet and do research together in real life. On behalf of all participants, the organizers want to express their deep gratitude to the entire Dagstuhl staff for their outstanding support and service accompanying this seminar. We further thank Jules Wulms for helping us collect the 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. In *Graph Drawing and Network Visualization (GD 2019)*, Lecture Notes in Computer Science. Springer, 2019. To appear.

- 4 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
- 5 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.
- 6 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer Verlag, 2013.
- 7 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.
- 8 F.V. Fomin, D. Lokshtanov, S. Saurabh, and M. Zehavi. *Kernelization: Theory of Parameterized Preprocessing*. Cambridge University Press, 2018.
- 9 Martin Grohe. Computing crossing numbers in quadratic time. *J. Comput. Syst. Sci.*, 68(2):285–302, 2004.
- 10 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.
- 11 Petr Hliněný 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.
- 12 Petr Hliněný and Abhisekh Sankaran. Exact crossing number parameterized by vertex cover. In *Graph Drawing and Network Visualization (GD 2019)*, *Lecture Notes in Computer Science*. Springer, 2019. To appear.
- 13 Michael Jünger and Petra Mutzel, editors. *Graph Drawing Software*. Springer, 2004.
- 14 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.
- 15 Fabian Klute and Martin Nöllenburg. Minimizing crossings in constrained two-sided circular graph layouts. In *Symposium on Computational Geometry (SoCG 2018)*, volume 99 of *LIPICs*, pages 53:1–53:14. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2018.
- 16 Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics and Its Applications. OUP Oxford, 2006.
- 17 Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing*, volume 12 of *Lecture Notes Series on Computing*. World Scientific, 2004.
- 18 Roberto Tamassia, editor. *Handbook on Graph Drawing and Visualization*. Chapman and Hall/CRC, 2013.
- 19 Paul Turán. A note of welcome. *Journal of Graph Theory*, 1(1):7–9, 1977.

2 Table of Contents

Executive Summary

Robert Ganian, Fabrizio Montecchiani, Martin Nöllenburg, and Meirav Zehavi . . . 82

Overview of Talks

Graph Drawing Extension Problems
Thekla Hamm 90

Planar insertion problems
Petr Hlinený 90

Constrained Embedding Problems
Ignaz Rutter 91

Graph Drawing beyond planarity and Parametrized Complexity
Michael Kaufmann 91

Open problems

Bundled Crossings
Steven Chaplick 92

Algorithmic and Combinatorial Applications of the Product Structure Theorems
Giordano Da Lozzo 93

Three open problems about orthogonal and upward drawings
Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani . 94

Parameterized Complexity of Computing Stack and Queue Numbers
Robert Ganian 97

Almost Separated Fixed Order Stack Layouts
Martin Gronemann 97

Embedding Upward Planar Graphs in two Pages
Martin Gronemann 98

Fine-grained complexity of the crossing number of almost planar graphs
Petr Hlinený 99

Labeling Curve Arrangements
Maarten Löffler 100

Bend Minimization in Orthogonal Drawings
Ignaz Rutter and Meirav Zehavi 103

Is Extending Partial Drawings of Level Planar Graphs FPT?
Ignaz Rutter 104

The Parameterized Complexity of Learning Small Decision Trees in Low-Dimensional Space
Manuel Sorge 105

Two open problems on drawings of complete graphs
Birgit Vogtenhuber 106

Working groups

Progress on Upward Planarity Testing <i>Robert Ganian, Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Chrysanthi Raftopoulou, and Kirill Simonov</i>	108
Progress on Embedding Upward Planar Graphs in two Pages <i>Michael A. Bekos, Giordano Da Lozzo, Fabrizio Frati, Martin Gronemann, and Chrysanthi Raftopoulou</i>	111
Progress on A Parameterized Approach to Orthogonal Compaction <i>Philipp Kindermann, Walter Didimo, Siddharth Gupta, Giuseppe Liotta, Alexander Wolff, and Meirav Zehavi</i>	113
Progress on Almost Separated Fixed Order Stack Layouts <i>Johannes Zink, Martin Gronemann, Thekla Hamm, Boris Klemz, Martin Nöllenburg, and Birgit Vogtenhuber</i>	116
Progress on Applications of the Product Structure Theorems <i>Giordano Da Lozzo, Michael A. Bekos, Petr Hlinený, and Michael Kaufmann</i>	118
Progress on the Parameterized Complexity of Small Decision Tree Learning <i>Stephen G. Kobourov, Maarten Löffler, Fabrizio Montecchiani, Raimund Seidel, Ignaz Rutter, Manuel Sorge, and Jules Wolms</i>	120
Participants	123
Remote Participants	123

3 Overview of Talks

3.1 Graph Drawing Extension Problems

Thekla Hamm (TU Wien, AT)

License  Creative Commons BY 4.0 International license
 Thekla Hamm

Joint work of Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, Martin Nöllenburg, Irene Parada, Birgit Vogtenhuber

Main reference Eduard Eiben, Robert Ganian, Thekla Hamm, Fabian Klute, Martin Nöllenburg: “Extending Partial 1-Planar Drawings”, in ICALP 2020, LIPIcs, Vol. 168, pp. 43:1–43:19, 2020.

URL <https://doi.org/10.4230/LIPIcs.ICALP.2020.43>

Main reference Robert Ganian, Thekla Hamm, Fabian Klute, Irene Parada, Birgit Vogtenhuber: “Crossing-Optimal Extension of Simple Drawings”, in ICALP 2021, LIPIcs, Vol. 198, pp. 72:1–72:17, 2021.

URL <https://doi.org/10.4230/LIPIcs.ICALP.2021.72>

The investigation of problems that ask for drawings of graphs with desirable properties (most commonly restricting crossings of edge drawings) while also fixing the drawing of a given subgraph is an increasingly popular direction in the field of graph drawing. These problems are also called *drawing extension problems*.

While the planar drawing extension problem can be solved in polynomial time, for many other important drawing styles, such as 1-planar, k -planar, IC-planar, straight-line planar and level planar, drawing extension is NP-hard. In this talk we explore the possibility of circumventing these hardness results when a large part of the graph is predrawn using the framework of parameterised complexity theory. In particular we review a general technique which can be used to show FPT results for a number of beyond-planar drawing styles and outline a variety of related open questions.

3.2 Planar insertion problems

Petr Hlinený (Masaryk University – Brno, CZ)

License  Creative Commons BY 4.0 International license
 Petr Hlinený

Joint work of Petr Hlinený, Markus Chimani, Gelasio Salazar

Main reference Markus Chimani, Petr Hlinený: “A tighter insertion-based approximation of the crossing number”, J. Comb. Optim., Vol. 33(4), pp. 1183–1225, 2017.

URL <https://doi.org/10.1007/s10878-016-0030-z>

A *planar insertion* problem is defined as follows: Given graphs G (planar) and H , the task of insertion of H into G is to find a crossing-minimal drawing of $G \cup H$ such that G itself is planar in the drawing. This problem is intermediate between ordinary crossing minimization and drawing extension problems, in the following sense. While in ordinary crossing minimization any drawing of the target graph is allowed, in planar insertion certain part of it (here G) must be planarly drawn. On the other hand, unlike in drawing extension problems, the planar part G may choose between its planar embeddings.

We survey past achievements in solving planar insertion problem variants. Firstly, we outline the linear-time algorithm for a single edge insertion by Gutwenger, Mutzel and Weiskircher from 2005, and show how this approximates the crossing number of a planar graph plus one edge (up to a multiplicative factor depending on the maximum degree). Note that determining the exact crossing number of a planar graph plus one edge is NP-hard by a result of Cabello and Mohar from 2011.

We then show how the multiple edge insertion problem can be in polynomial time approximated up to an additive error depending on the number of inserted edges and the maximum degree. Again, the general question is NP-hard. From another perspective, we show that the multiple edge insertion problem can be solved exactly in FPT time when the parameter is the number of inserted edges.

3.3 Constrained Embedding Problems

Ignaz Rutter (*Universität Passau, DE*)

License  Creative Commons BY 4.0 International license
© Ignaz Rutter

Determining a planar embedding of a graph is a classical problem. In many applications, one is interested in finding a planar embedding that satisfies additional constraints. In this talk, we survey several techniques and demonstrate their application on a number such problems. For local constraints that mostly concern rotations, i.e., the circular orders of edges around vertices, PQ-trees and their circular variants known as PC-trees serve as a powerful tool. If a more global view of the possible embeddings is necessary, often the SPQR-tree is useful, as it breaks up the complicated choice of a planar embedding into several simple and independent choices. Lastly, the ability to synchronize the rotations of different vertices is a powerful method, whose solution requires a combination of both of the above techniques.

3.4 Graph Drawing beyond planarity and Parametrized Complexity

Michael Kaufmann (*Universität Tübingen, DE*)

License  Creative Commons BY 4.0 International license
© Michael Kaufmann

In this talk, we gave an overview on different aspects on graph drawing beyond planarity, i.e. drawings where some crossing configurations for the edges are forbidden. Notable criteria are density of the graphs, recognition, class hierarchies, constraints,

We discussed several results from the literature related to aspects of parametrized complexity, in particular kernel-based methods, separators, path – and treewidth- related questions. We reviewed the most important results from the seminal paper on the parametrized complexity of 1-planarity by Bannister, Cabello and Eppstein [1]. Furthermore we highlighted some of the methods developed on track-layout of fan-planar graphs by Biedl et al. [3].

We extracted and discuss possible open directions related to k -planarity, fan-planarity and other classes of beyond-planar graphs that could be attacked during and after the workshop.

A notable paper which we did not included is the work by Bhore et al. [2], which extends the recent research direction on linear layouts towards parametrized complexity.

References

- 1 Michael J. Bannister, Sergio Cabello, and David Eppstein. Parameterized complexity of 1-planarity. In *Algorithms and Data Structures – 13th International Symposium, WADS 2013, London, ON, Canada, August 12–14, 2013. Proceedings*, volume 8037 of *Lecture Notes in Computer Science*, pages 97–108. Springer, 2013.
- 2 Sujoy Bhore, Robert Ganian, Fabrizio Montecchiani, and Martin Nöllenburg. Parameterized algorithms for book embedding problems. *J. Graph Algorithms Appl.*, 24(4):603–620, 2020.
- 3 Therese C. Biedl, Steven Chaplick, Michael Kaufmann, Fabrizio Montecchiani, Martin Nöllenburg, and Chrysanthi N. Raftopoulou. Layered fan-planar graph drawings. In *45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, August 24–28, 2020, Prague, Czech Republic*, volume 170 of *LIPICs*, pages 14:1–14:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

4 Open problems

4.1 Bundled Crossings

Steven Chaplick (Maastricht University, NL)

License  Creative Commons BY 4.0 International license
© Steven Chaplick

An effective way to reduce clutter in a graph drawing that has (many) crossings is to group edges that travel in parallel into *bundles*. This concept was introduced by Holten [4].

Each edge can participate in many such bundles. Any crossing in this bundled drawing occurs between two bundles (possibly one such bundle will consist of a single edge) and these crossings are referred to as *asbundled crossing*. We consider the problem of bundled crossing minimization: A graph is given and the goal is to find a bundled drawing with at most k bundled crossings. This problem is known to be NP-complete when in both the case when a simple drawing is required and when the drawing is allowed to be non-simple. The latter (non-simple) case turns out to be equivalent to the computing the graph genus [1], and as such has a long history including efficient FPT algorithms, see, e.g., whereas for the simple case it is open whether the problem is FPT [5]. In the case of simple drawings the problem is known to be FPT when one further insists on a *circular layout* where vertices are placed in convex position and all edges are required to be drawn within the convex hull of the vertices [2].

Finally, we note that even when given a graph drawn in the plane (with crossings) and parameter k , and one desires to bundle this drawing to have at most k crossings, the problem is also NP-complete [3]. In other words, trying to find an optimal bundling of a given drawing is also an interesting problem where, as far as we are aware, the question of fixed-parameter tractability remains open as well.

References

- 1 Md. Jawaherul Alam, Martin Fink, and Sergey Pupyrev. The bundled crossing number. In Yifan Hu and Martin Nöllenburg, editors, *GD*, volume 9801 of *LNCS*, pages 399–412. Springer, 2016.
- 2 Steven Chaplick, Thomas C. van Dijk, Myroslav Kryven, Ji-won Park, Alexander Ravsky, and Alexander Wolff. Bundled crossings revisited. *J. Graph Algorithms Appl.*, 24(4):621–655, 2020.
- 3 Martin Fink, John Hershberger, Subhash Suri, and Kevin Verbeek. Bundled crossings in embedded graphs. In Evangelos Kranakis, Gonzalo Navarro, and Edgar Chávez, editors, *LATIN*, volume 9644 of *LNCS*, pages 454–468. Springer, 2016.
- 4 Danny Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Trans. Vis. Comput. Graphics*, 12(5):741–748, 2006.
- 5 Ken-ichi Kawarabayashi, Bojan Mohar, and Bruce A. Reed. A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width. In *FOCS*, pages 771–780. IEEE, 2008.

4.2 Algorithmic and Combinatorial Applications of the Product Structure Theorems

Giordano Da Lozzo (University of Rome III, IT)

License  Creative Commons BY 4.0 International license
© Giordano Da Lozzo

Main reference Vida Dujmović, Gwenaël Joret, Piotr Micek, Pat Morin, Torsten Ueckerdt, David R. Wood: “Planar Graphs Have Bounded Queue-Number”, *J. ACM*, Vol. 67(4), pp. 22:1–22:38, 2020.

URL <https://dl.acm.org/doi/10.1145/3385731>

Consider two graphs A and B . The *strong product* of A and B , denoted by $A \boxtimes B$, is the graph such that: (i) $V(A \boxtimes B) = V(A) \times V(B)$ and (ii) there exists an edge between the vertices $(a_1, b_1), (a_2, b_2) \in V(A \boxtimes B)$ if and only if one of the following occurs: (a) $a_1 = a_2$ and $b_1 b_2 \in E(B)$, (b) $b_1 = b_2$ and $a_1 a_2 \in E(A)$, or $a_1 a_2 \in E(A)$ and $b_1 b_2 \in E(B)$. In a breakthrough result, Dujmović et al. [1] have shown that every planar graph is a subgraph of the strong product of a graph of treewidth 8 and a path. In the same paper, such a result has also been generalized to graphs of bounded Euler genus and to proper minor-closed classes of graphs. In a recent preprint [4], Dujmović, Morin, and Wood have extended this result to some non-minor-closed graph classes. In particular, they proved that every k -planar graph is a subgraph of the strong product of a graph of treewidth $O(k^5)$ and a path. These results, commonly referred to as the Product Structure Theorems (PSTs), have proved essential to solve several *combinatorial* long-standing open questions for the above mentioned graph classes. For instance, the PSTs allowed to prove that planar graphs have bounded queue number and bounded non-repetitive chromatic number [1], to improve the best known bounds for p -centered colorings of planar graphs and graphs excluding any fixed graph as a subdivision [2], to find shorter adjacency labelings of planar graphs [5], and to find asymptotically optimal adjacency labelings of planar graphs [3].

First, we suggest to keep exploring the above line of research by studying the following problem.

OP1: Can the PST be improved for k -planar graphs, with $k \in \{1, 2\}$?

Furthermore, it is interesting to consider a new line of research aimed at investigating the *algorithmic applications* of the PSTs. We believe that these theorems could support new results in fixed-parameter tractability, approximations, and bidimensionality theory. In particular, we propose the following problem.

OP2: Are there notable applications of the PST for topological k -planar graphs to obtain FPT algorithms parameterized by k ?

References

- 1 V. Dujmović, G. Joret, P. Micek, P. Morin, T. Ueckerdt, and D. Wood. Planar Graphs Have Bounded Queue-Number. *J. ACM*, 67(4): 22:1–22:38 (2020).
- 2 M. Debski, S. Felsner, P. Micek, and F. Schröder. Improved bounds for centered colorings. *SODA 2020*: 2212–2226 (2020).
- 3 V. Dujmović, L. Esperet, C. Gavaille, G. Joret, P. Micek, and P. Morin. Adjacency Labelling for Planar Graphs (and Beyond). *FOCS 2020*: 577–588 (2020).
- 4 V. Dujmović, P. Morin, and D. Wood. Graph product structure for non-minor-closed classes. *CoRR* abs/1907.05168 (2020).
- 5 M. Bonamy, C. Gavaille, and M. Pilipeczuk. Shorter Labeling Schemes for Planar Graphs. *SODA 2020*: 446–462 (2020).

4.3 Three open problems about orthogonal and upward drawings

Emilio Di Giacomo (University of Perugia, IT), Walter Didimo (University of Perugia, IT), Giuseppe Liotta (University of Perugia, IT), and Fabrizio Montecchiani (University of Perugia, IT)

License © Creative Commons BY 4.0 International license
© Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani

Problem 1: Rectilinear planarity testing

A graph is *planar* if it admits a drawing in the plane such that edges intersect only at common endpoints. Testing graph planarity is a fundamental problem in graph algorithms that have been studied in several variants and restrictions, such as upward planarity, clustered planarity, and constrained planarity. A classical planarity variant is the *rectilinear planarity*, which asks whether a planar graph with maximum vertex degree four admits a *rectilinear drawing*, i.e., a planar drawing where each edge is either a horizontal or a vertical segment.

Rectilinear drawings are a special case of *orthogonal drawings*, where edges are represented as chains of horizontal and vertical segments. Orthogonal drawings are among the most investigated research subjects in graph drawing (see, e.g., [6, 12]). A natural measure of the complexity of an orthogonal drawing is the number of bends along the edges, which should be minimized. In this sense, a rectilinear drawing is optimal, since it has no bends.

Garg and Tamassia [13] proved that rectilinear planarity testing is NP-complete. In fact, it is even NP-hard to approximate the minimum number of bends in an orthogonal drawing with an $O(n^{1-\varepsilon})$ error for any $\varepsilon > 0$ [13]. On the other hand if the input graph is *plane*, i.e., it has a fixed embedding in the plane, Tamassia [19] showed that rectilinear planarity testing can be decided in polynomial time. When a planar embedding is not given as part of the input, polynomial-time algorithms exist for some restricted cases, such as subcubic planar graphs and series-parallel graphs [5, 7, 11, 18, 20]).

Given the hardness results for rectilinear planarity testing, it is natural to study its parameterized complexity. Few results are known in this direction: Didimo and Liotta [10] described an algorithm for biconnected planar graphs that runs in $O(6^r n^4 \log n)$ time, where r is the number of degree-4 vertices. More recently Di Giacomo, Liotta, and Montecchiani [8] proved that the problem belongs to the XP class when parameterized by the treewidth and to the FPT class when parameterized by the treewidth plus the number of vertices of degree at most 2.

In the light of these last results it is natural to ask if the problem is in FPT when parameterized by only one of the two parameters.

► **Problem 1.** Is rectilinear planarity testing in FPT when parameterized by the treewidth? Is rectilinear planarity testing in FPT when parameterized by the number of degree-2 vertices?

Problem 2: Orthogonal compaction

As mentioned above, if the planar embedding is fixed, an orthogonal drawing with the minimum number of bends can be computed in polynomial time. Thus, one of the most used algorithmic frameworks to compute orthogonal drawings is the one proposed by Tamassia [19], usually referred to as the *Topology-Shape-Metrics approach*. This approach works in three steps. The first step, called *Planarization*, fixes the *topology* of the input graph G , that is, it computes a planar embedding of G ; if G is not planar a planarization of G is constructed, i.e., a planar graph obtained by replacing crossings with dummy vertices; the optimization goal of this step is to reduce the number of crossings and therefore of dummy vertices. The second

step, called *Orthogonalization*, decides the *shape* of the drawing, that is, it computes what is called an *orthogonal representation* of G . An orthogonal representation is a description of the shape of an orthogonal drawing in terms of the angles at the vertices and the number of bends along the edges. In this step the optimization goal is to minimize the number of bends, which, as said above, can be done in polynomial time once the planar embedding is fixed. In the third step the actual coordinates of the vertices and bends are decided thus fixing the *metrics* of the drawing. This step is called *Compaction* step because the coordinates are assigned with the goal of minimizing the area of the drawing (or the total length of the edges).

The compaction step hence, solves the following problem, called the *orthogonal compaction* problem: Given an orthogonal representation, compute vertex and bend coordinates in such a way that the area is minimized. This problem is known to be NP-complete [17] but it is polynomially-time solvable for *turn-regular* orthogonal representations [3]. An orthogonal representation is turn-regular if it does not contain any pairs of *kitty corners*. A pair of kitty corners is a pair of vertices u and v such that: (i) both u and v form a $\frac{3\pi}{2}$ angle inside a face f ; and (ii) walking clockwise along the boundary of f from u (included) to v (excluded) or vice versa the number of encountered vertices that form an angle of $\frac{\pi}{2}$ minus the number of encountered vertices that form an angle of $\frac{3\pi}{2}$ is 2. The two mentioned results suggest the following problem.

► **Problem 2.** Is orthogonal compaction problem in FPT when parameterized by the number of kitty corners?

Problem 3: Upward planarity testing

Upward planarity is another variant of planarity that has been widely investigated in the literature. An *upward planar drawing* of a directed acyclic graph is a planar drawing such that all edges are drawn as curves monotonically increasing in the upward direction. Similar to the case of rectilinear and orthogonal planarity, the problem is polynomially time solvable if a planar embedding of the input graph is fixed [1] and it is NP-complete if the planar embedding can be changed [13]. In the variable embedding setting polynomial-time algorithms exist for special cases, such as outerplanar DAGs [16] or series-parallel DAGs [9]. In particular, the problem can be solved in polynomial time when the input DAG has a single source, i.e., a single vertex without incoming edges [2, 15]. These results naturally motivate the following problem.

► **Problem 3.** Is upward planarity testing in FPT when parameterized by the number of sources?

It is worth mentioning that FPT algorithms exist for the upward planarity testing when parameterized by the number of cut-vertices and the number of triconnected components [4], only by the number of triconnected components [14], by the difference between the number of edges and the number of vertices [14] and by the number of triconnected components and the diameter of any split component [9].

References

- 1 P. Bertolazzi, G. D. Battista, G. Liotta, and C. Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 12(6):476–497, 1994. doi:10.1007/BF01188716.
- 2 P. Bertolazzi, G. D. Battista, C. Mannino, and R. Tamassia. Optimal upward planarity testing of single-source digraphs. *SIAM J. Comput.*, 27(1):132–169, 1998. doi:10.1137/S0097539794279626.
- 3 S. S. Bridgeman, G. D. Battista, W. Didimo, G. Liotta, R. Tamassia, and L. Vismara. Turn-regularity and optimal area drawings of orthogonal representations. *Comput. Geom.*, 16(1):53–93, 2000. doi:10.1016/S0925-7721(99)00054-1.

- 4 H. Y. Chan. A parameterized algorithm for upward planarity testing. In S. Albers and T. Radzik, editors, *Algorithms – ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings*, volume 3221 of *Lecture Notes in Computer Science*, pages 157–168. Springer, 2004. doi:10.1007/978-3-540-30140-0_16.
- 5 Y. Chang and H. Yen. On bend-minimized orthogonal drawings of planar 3-graphs. In *SOCG 2017*, volume 77 of *LIPICs*, pages 29:1–29:15. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2017.
- 6 G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing*. Prentice-Hall, 1999.
- 7 G. Di Battista, G. Liotta, and F. Vargiu. Spirality and optimal orthogonal drawings. *SIAM J. Comput.*, 27(6):1764–1811, 1998.
- 8 E. Di Giacomo, G. Liotta, and F. Montecchiani. Sketched representations and orthogonal planarity of bounded treewidth graphs. In D. Archambault and C. D. Tóth, editors, *Graph Drawing and Network Visualization – 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 379–392. Springer, 2019. doi:10.1007/978-3-030-35802-0_29.
- 9 W. Didimo, F. Giordano, and G. Liotta. Upward spirality and upward planarity testing. *SIAM J. Discret. Math.*, 23(4):1842–1899, 2009. doi:10.1137/070696854.
- 10 W. Didimo and G. Liotta. Computing orthogonal drawings in a variable embedding setting. In *ISAAC 1998*, volume 1533 of *LNCS*, pages 79–88. Springer, 1998.
- 11 W. Didimo, G. Liotta, and M. Patrignani. Bend-minimum orthogonal drawings in quadratic time. In *GD 2018*, volume 11282 of *LNCS*, pages 481–494. Springer, 2018.
- 12 C. A. Duncan and M. T. Goodrich. Planar orthogonal and polyline drawing algorithms. In *Handbook of Graph Drawing and Visualization*, pages 223–246. Chapman and Hall/CRC, 2013.
- 13 A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.
- 14 P. Healy and K. Lynch. Two fixed-parameter tractable algorithms for testing upward planarity. *Int. J. Found. Comput. Sci.*, 17(5):1095–1114, 2006. doi:10.1142/S0129054106004285.
- 15 M. D. Hutton and A. Lubiw. Upward planning of single-source acyclic digraphs. *SIAM J. Comput.*, 25(2):291–311, 1996. doi:10.1137/S0097539792235906.
- 16 A. Papakostas. Upward planarity testing of outerplanar dags. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing, DIMACS International Workshop, GD '94, Princeton, New Jersey, USA, October 10-12, 1994, Proceedings*, volume 894 of *Lecture Notes in Computer Science*, pages 298–306. Springer, 1994. doi:10.1007/3-540-58950-3_385.
- 17 M. Patrignani. On the complexity of orthogonal compaction. *Comput. Geom.*, 19(1):47–67, 2001. doi:10.1016/S0925-7721(01)00010-4.
- 18 M. S. Rahman, N. Egi, and T. Nishizeki. No-bend orthogonal drawings of subdivisions of planar triconnected cubic graphs. *IEICE Transactions*, 88-D(1):23–30, 2005.
- 19 R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comp.*, 16(3):421–444, 1987.
- 20 X. Zhou and T. Nishizeki. Orthogonal drawings of series-parallel graphs with minimum bends. *SIAM J. Discrete Math.*, 22(4):1570–1604, 2008.

4.4 Parameterized Complexity of Computing Stack and Queue Numbers

Robert Ganian (TU Wien, AT)

License  Creative Commons BY 4.0 International license
© Robert Ganian

The problems of computing a queue or stack layout with the minimum number of pages (the so-called QUEUE NUMBER and STACK NUMBER problems) are well-studied and known to be NP-complete, but we still do not understand the conditions under which these problems become tractable. In particular, while recent works have shown that both problems are fixed-parameter tractable when parameterized by the vertex cover number [1, 2], we do not know anything about their parameterized complexity when parameterized by clique-width, treewidth, pathwidth, treedepth, feedback vertex number, and even feedback edge number. In fact, we do not even know whether the problem is fixed-parameter tractable, W-hard, or paraNP-hard when parameterized by a parameter as simple as the edge deletion distance to a collection of paths.

References

- 1 Sujoy Bhore and Robert Ganian and Fabrizio Montecchiani and Martin Nöllenburg. *Parameterized Algorithms for Book Embedding Problems*. J. Graph Algorithms Appl., vol. 24, issue 4, 2020.
- 2 Sujoy Bhore and Robert Ganian and Fabrizio Montecchiani and Martin Nöllenburg. *Parameterized Algorithms for Queue Layouts*. Graph Drawing and Network Visualization – 28th International Symposium, GD 2020, Vancouver, BC, Canada, September 16-18, 2020, Revised Selected Papers.

4.5 Almost Separated Fixed Order Stack Layouts

Martin Gronemann (Universität Osnabrück, DE)

License  Creative Commons BY 4.0 International license
© Martin Gronemann

Book embeddings have a long history in graph theory. Today, book embeddings are often referred to as stack layouts. Formally, a stack layout consists of a linear ordering of the vertices σ drawn on a line and a partitioning of the edges into *pages* such that no two edges on the same page cross when drawn in the same half-plane defined by the line. Given a graph G , the minimum number of pages required in any stack layout of G is referred to as stack number $\text{sn}(G)$. Determining the stack number of a graph is inherently difficult. While this problem is linear time solvable for $\text{sn}(G) = 1$ by testing if the input graph is outerplanar, testing if two stacks are sufficient is already NP-complete [2].

Therefore, it makes sense to consider a more restricted variant of this problem by assuming that the vertex order σ is given as part of the input. Hence, it remains to assign the edges to pages by using as few pages as possible. This problem is sometimes referred to as the FIXED-ORDER BOOK THICKNESS problem. Unfortunately, also this problem is known to be NP-complete for four or more pages [1]. However, for some vertex orderings, the problem becomes easier. Consider a bipartite graph with partitions A and B . If in the vertex order A and B are *separated*, that is, all vertices of A precede those of B , the stack number equals the number of pairwise crossing edges. One may now generalize this concept of being separated by assuming that for a bipartite graph $G = (A \cup B, E)$ the vertices of A and B form k

consecutive blocks in the fixed vertex order. More, specifically, in σ there are exactly $\frac{k}{2}$ consecutive blocks containing vertices of A and $\frac{k}{2}$ consecutive blocks containing solely vertices of B . We refer to such a layout as *fixed k -separated layout*.

Open Problem. Is the FIXED-ORDER BOOK THICKNESS problem for fixed k -separated layouts fixed-parameter tractable in k ?

References

- 1 Walter Unger. On the k -colouring of circle-graphs. In Robert Cori and Martin Wirsing, editors, *STACS 88*, pages 61–72, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.
- 2 Avi Wigderson. The complexity of the Hamiltonian circuit problem for maximal planar graphs. Technical Report TR-298, EECS Department, Princeton University, 1982. [arXiv: https://www.math.ias.edu/avi/node/820](https://www.math.ias.edu/avi/node/820).

4.6 Embedding Upward Planar Graphs in two Pages

Martin Gronemann (Universität Osnabrück, DE)

License  Creative Commons BY 4.0 International license
© Martin Gronemann

Book embeddings of graphs are a classic topic in graph theory and graph drawing [1, 2, 4, 3, 6, 7, 14, 15]. Formally, in a *book embedding*, the vertices of a given graph must be ordered along a line, called *spine*, and its edges must be drawn in different half-planes bounded by the spine, called *pages* of the book, such that no two edges of the same page cross. For a 2-page book embedding, where only two pages are available, one can use the two half-planes defined by the spine for drawing the edges of the graph [14]. As a consequence such a drawing is planar.

For directed graphs (digraphs), Heath, Pemmaraju, and Trenk introduced a variant of book embeddings, called *upward*, in which all the edges are oriented in the upward direction, i.e., such that for every directed edge uv of the given graph u precedes v along the spine [10]. Clearly, this immediately implies that the input graph is acyclic. In the case, in which only two pages are available, one obtains a special form of an upward planar drawing. An *upward planar* drawing of a directed acyclic graph is a planar drawing in which each edge uv is drawn as a y -monotone curve from u to v . A planar directed acyclic graph that admits such a drawing is called *upward planar*. However, deciding whether a planar directed acyclic graph is upward planar is known to be NP-complete [8]. An important family of graphs in this context are the *st-planar graphs*, i.e., planar directed acyclic graphs having only one source s and one sink t . It is known that all *st-planar* graphs are upward planar and that every upward planar graph is a subgraph of an *st-planar* graph. An interesting question which attracted attention in the literature is the 2-page embeddability of *st-planar* graphs [9, 10]. For specific families of *st-planar* graphs or graphs where certain conditions are met, the existence of an upward 2-page book embedding can be efficiently decided [5, 12, 11]. However, the general question remains unanswered [13].

Open Problem. What is the complexity of deciding whether a given embedded *st-planar* graph admits a (planar) upward 2-page book embedding?

References

- 1 M. A. Bekos, T. Bruckdorfer, M. Kaufmann, and C. N. Raftopoulou. The book thickness of 1-planar graphs is constant. *Algorithmica*, 79(2):444–465, 2017.

- 2 M. A. Bekos, G. Da Lozzo, S. Griesbach, M. Gronemann, F. Montecchiani, and C. N. Raftopoulou. Book embeddings of nonplanar graphs with small faces in few pages. In S. Cabello and D. Z. Chen, editors, *SoCG 2020*, volume 164 of *LIPICs*, pages 16:1–16:17. Schloss Dagstuhl, 2020.
- 3 M. A. Bekos, M. Gronemann, and C. N. Raftopoulou. Two-page book embeddings of 4-planar graphs. *Algorithmica*, 75(1):158–185, 2016.
- 4 F. Bernhart and P. C. Kainen. The book thickness of a graph. *Journal of Combinatorial Theory, Series B*, 27(3):320 – 331, 1979.
- 5 C. Binucci, G. Da Lozzo, E. D. Giacomo, W. Didimo, T. Mchedlidze, and M. Patrignani. Upward book embeddings of st-graphs. In G. Barequet and Y. Wang, editors, *SoCG 2019*, volume 129 of *LIPICs*, pages 13:1–13:22. Schloss Dagstuhl, 2019.
- 6 F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg. Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM Journal on Algebraic Discrete Methods*, 8(1):33–58, 1987.
- 7 H. Enomoto, T. Nakamigawa, and K. Ota. On the pagenumber of complete bipartite graphs. *Journal of Combinatorial Theory, Series B*, 71(1):111–120, 1997.
- 8 A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.
- 9 L. S. Heath and S. V. Pemmaraju. Stack and queue layouts of posets. *SIAM Journal on Discrete Mathematics*, 10(4):599–625, 1997.
- 10 L. S. Heath, S. V. Pemmaraju, and A. N. Trenk. Stack and queue layouts of directed acyclic graphs: Part I. *SIAM Journal on Computing*, 28(4):1510–1539, 1999.
- 11 T. Mchedlidze and A. Symvonis. Crossing-free acyclic hamiltonian path completion for planar st-digraphs. In Y. Dong, D. Du, and O. H. Ibarra, editors, *ISAAC 2009*, volume 5878 of *LNCS*, pages 882–891. Springer, 2009.
- 12 T. Mchedlidze and A. Symvonis. Crossing-optimal acyclic HP-completion for outerplanar st-digraphs. *JGAA*, 15(3):373–415, 2011.
- 13 R. Nowakowski and A. Parker. Ordered sets, pagenumbers and planarity. *Order*, 6(3):209–218, 1989.
- 14 A. Wigderson. The complexity of the Hamiltonian circuit problem for maximal planar graphs. Technical report, 298, EECS Department, Princeton University, 1982.
- 15 M. Yannakakis. Embedding planar graphs in four pages. *Journal of Computer and System Sciences*, 38(1):36–67, 1989.

4.7 Fine-grained complexity of the crossing number of almost planar graphs

Petr Hlinený (Masaryk University – Brno, CZ)

License © Creative Commons BY 4.0 International license
© Petr Hlinený

A graph is *almost planar* (or near-planar) if it becomes planar after deleting a suitable one edge. Cabello and Mohar in 2010 [1] proved that, surprisingly, computing the exact crossing number of almost planar graphs is NP-hard. At the same time this problem can be efficiently approximated, up to the factor of maximum degree, by a planar edge insertion solution. Specially, for cubic almost planar graphs, the mentioned edge insertion solves the crossing number exactly. We hence suggest to investigate the possibility of having an FPT algorithm for the exact crossing number of almost planar graphs parameterized by the maximum degree.

Furthermore, one can modify the hardness reduction of Cabello and Mohar in a way that it uses only 16 vertices of degree greater than 3 (this is not published, but it follows from a 2015 paper by Hliněný and Salazar [2] on hardness of joint crossing number). Therefore, it would be interesting to determine the smallest $h > 0$ such that computing the exact crossing number of almost planar graphs with only h vertices of degree greater than 3 is NP-hard (as we know that $h \leq 16$).

References

- 1 S. Cabello and B. Mohar. Adding one edge to planar graphs makes crossing number hard. In *SoCG*, pages 68–76. ACM, 2010.
- 2 P. Hliněný and G. Salazar. On Hardness of the Joint Crossing Number. In *ISAAC*, volume 9472 of *LNCS*, pages 603–613. Springer, 2015.

4.8 Labeling Curve Arrangements

Maarten Löffler (Utrecht University, NL)

License  Creative Commons BY 4.0 International license
© Maarten Löffler

Introduction. Consider the following problem. Given is a region in the plane (say, a polygon, or a collection of polygons), together with a set of curves that lie in the interior of the region, and which start and end on the boundary of the region. Refer to Figure 1a. Now suppose we wish to annotate these curves with some text describing the meaning of the curves. One option is to write the text along the curve itself (*interior labeling*), but in some applications this is undesirable, as the text might obfuscate other important information. In this case, we may choose to instead extend the curves outside the region, and label (one or both sides of) the curve there. Refer to Figure 1b. When doing this, we have a choice: we can extend each curve on either side. Depending on these choices, we may reach a conflicting labeling (where several labels overlap each other) or not. Refer to Figure 1c. Furthermore, in order to avoid conflicts, we might extend a curve on both sides (and label each side of the curve on another end), or we might extend a curve even farther to move the text away from the region.

This problem was recently studied in the context of *nonogram* generation [1]. A *curved nonogram* is a variation on the classic logic puzzle in which the objective is to colour several cells in an arrangement of curves based on a sequence of *clues*, which are placed outside the diagram [2]. When placing these labels naïvely, conflicts may occur. Refer to Figure 2. Löffler and Nöllenburg show that in general, the problem of finding a non-conflicting labeling of a curve arrangement is NP-hard, but they provide polynomial-time solutions for several restricted settings.

Open Problem. The results from [1] suggest that, while hard in general, the problem may be easy when certain *parameters* are small. Depending on the application, several natural parameters come to mind.

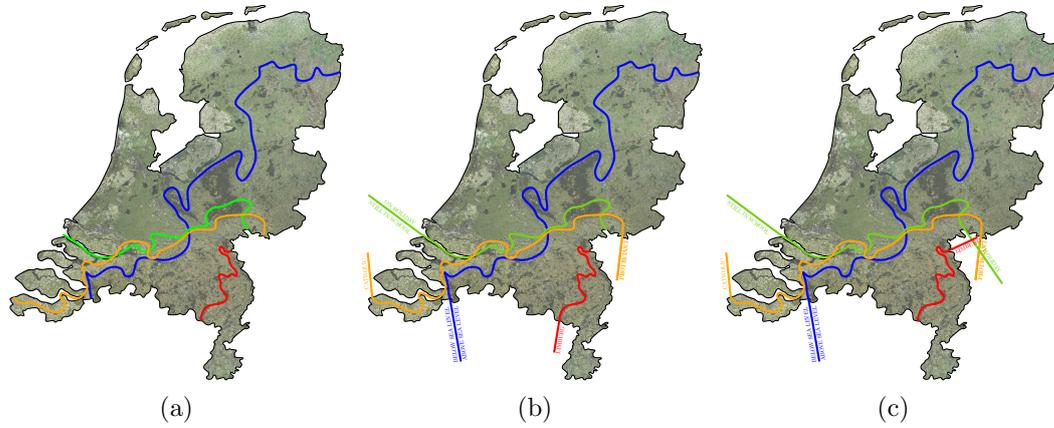
Formally, we may define the input to the curve arrangement labeling problem as:

- a polygon P ;
- n pairs of ports on P ;
- up to $2n$ label sizes.

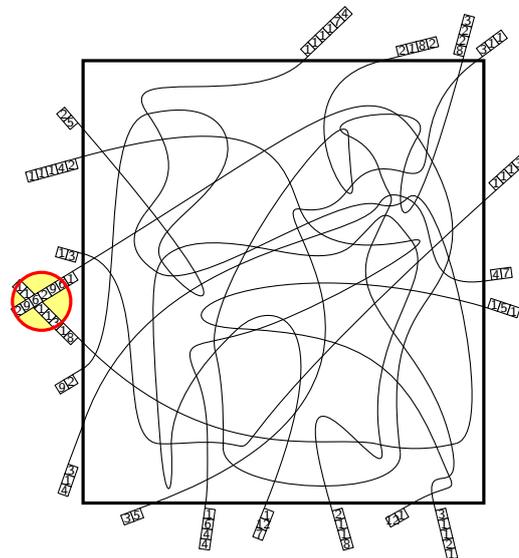
The output is then:

- a location of each label;
- a curve from each label to one of the ports.

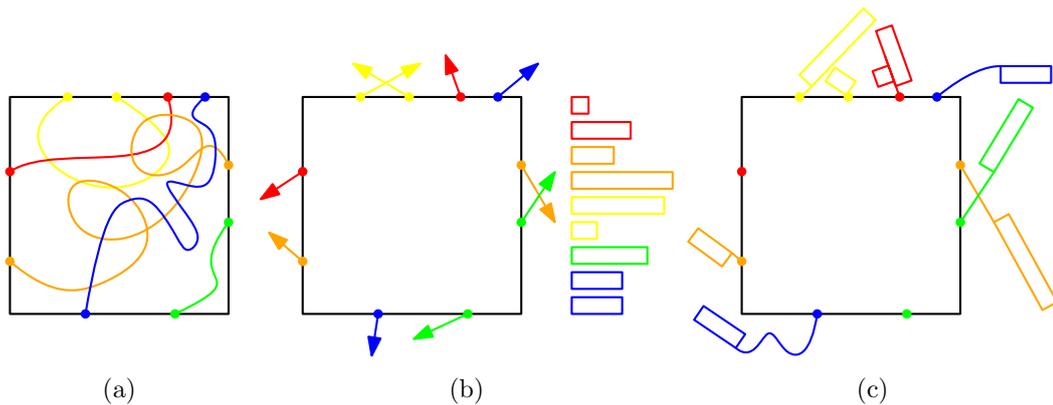
Refer to Figure 3.



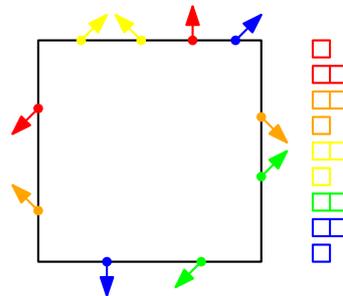
■ **Figure 1** (a) An arrangement of curves in a polygon. (b) A labeling of some of the curves. Some curves may be labeled only on one side. For curves which are labeled on both sides, we can either place both labels on the same end, or on opposite ends. (c) Some possible locations for curve labels may conflict each other.



■ **Figure 2** One application of the curve arrangement labeling problem is in automatic puzzle generation.



■ **Figure 3** (a) An arrangement of curves in a polygon. (b) The resulting labeling problem. The interior is irrelevant; only the tangent vectors of the curves at the ports are retained. (c) A possible solution.



■ **Figure 4** We may restrict the problem in several ways.

The open problem we propose is to investigate the parameterized complexity of the curve arrangement labeling problem. We suggest several possible parameters, which we may classify into *input parameters* (which quantify certain aspects about the problem input) and *output parameters* (which restrict the set of labelings considered).

Possible input parameters include:

- the number of port orientations;
- the maximum label length;
- the complexity of polygon.

Refer to Figure 4.

Possible output parameters include:

- the number of unplaced labels;
- the number of extended labels;
- the maximum extension length;
- the complexity of the extensions;
- the number of outside crossings;
- the number of split labels;
- the size of the bounding box.

References

- 1 Maarten Löffler and Martin Nöllenburg. Labeling nonograms. In *Proc. 36st European Workshop on Computational Geometry*, pages 53:1–8, 2020.

- 2 Mees van de Kerkhof, Tim de Jong, Raphael Parment, Maarten Löffler, Amir Vaxman, and Marc van Kreveld. Design and automated generation of japanese picture puzzles. In *Proc. 40th Annual Conference of the European Association for Computer Graphics*, 2019.

4.9 Bend Minimization in Orthogonal Drawings

Ignaz Rutter (*Universität Passau, DE*) and Meirav Zehavi (*Ben-Gurion University, IL*)

License  Creative Commons BY 4.0 International license
© Ignaz Rutter and Meirav Zehavi

Let $G = (V, E)$ be a graph with maximum degree 4. In an planar orthogonal drawing of G the vertices are mapped to grid points and the edges are mapped to pairwise non-crossing chains of horizontal and vertical segments that connect the endpoints of each edge. A bend is an interior point of an edge, where a horizontal and a vertical segment meet. Minimizing the number of bends in planar orthogonal drawings is a classical problem. If the graph comes with a fixed combinatorial embedding, then the number of bends can be minimized efficiently [5], whereas without a fixed combinatorial embedding, it is even NP-complete to decide whether there exists a bend-free planar orthogonal drawing [4].

In an attempt to work around the NP-hardness and to gain more control of the drawing, Bläsius et al. introduced two variants of the problem. In FLEXDRAW the input graph $G = (V, E)$ comes together with a flexibility function $f: E \rightarrow \mathbb{N}_0 \cup \{\infty\}$, which assigns to each edge e a flexibility. The question is whether there exists an orthogonal planar drawing such that each edge e has at most $f(e)$ bends. The problem can be solved in polynomial time if $f(e) > 0$ holds for all $e \in E[1]$ and it is FPT with respect to the number of edges with $f(e) = 0$ [2].

The disadvantage of these approaches is that, in the negative case, the algorithm does not output any drawing. To remedy this, Bläsius et al. [3] introduce OPTIMAL FLEX DRAW, whose input consists of a graph $G = (V, E)$ and for each edge $e \in E$ a cost function $c_e: \mathbb{N}_0 \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ that specifies for each edge a cost function. The cost of a drawing is then $\sum_{e \in E} c_e(b_e)$, where b_e denotes the number of bends in the drawing. They show that an optimal drawing can be found efficiently, if (i) all cost functions are convex and (ii) $c_e(1) = 0$ for all $e \in E$, i.e., the first bend on each edge is free.

Our question is whether OPTIMAL FLEX DRAW is FPT w.r.t. k if (i) all cost functions are convex and (ii) all but k edges $e \in E$ satisfies $c_e(1) = 0$.

As a response to the above open problem, Meirav Zehavi posed the question whether a similar model could work for finding a drawing that optimizes the number of crossings on graphs that are not necessarily planar, or when no planar drawing is given. Since both problems are in the same spirit, this new problem was called OPTIMAL FLEX CROSSING.

References

- 1 Thomas Bläsius, Marcus Krug, Ignaz Rutter, and Dorothea Wagner. Orthogonal graph drawing with flexibility constraints. *Algorithmica*, 68(4):859–885, 2014.
- 2 Thomas Bläsius, Sebastian Lehmann, and Ignaz Rutter. Orthogonal graph drawing with inflexible edges. *Comput. Geom.*, 55:26–40, 2016.
- 3 Thomas Bläsius, Ignaz Rutter, and Dorothea Wagner. Optimal orthogonal graph drawing with convex bend costs. *ACM Trans. Algorithms*, 12(3):33:1–33:32, 2016.
- 4 Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.
- 5 Roberto Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.

4.10 Is Extending Partial Drawings of Level Planar Graphs FPT?

Ignaz Rutter (Universität Passau, DE)

License  Creative Commons BY 4.0 International license
 © Ignaz Rutter

A (k -)level graph is a directed graph $G = (V, E)$ together with a leveling $\ell: V \rightarrow \{1, \dots, k\}$ such that each directed edge (u, v) satisfies $\ell(u) < \ell(v)$. A level drawing of G is a drawing of G where each edge is drawn as a y -monotone curve and each vertex v in V lies on the horizontal line with $y = \ell(v)$. Such a drawing is level planar, if its edges do not cross, except at common endpoints. A level graph is level planar if it admits a level planar drawing.

Level-planarity has been an active topic of research and it is well-known that level-planar graphs can be recognized in polynomial time. In fact, there are several algorithms that run in quadratic time [6, 3, 2], and even a linear-time algorithm is known [5, 4]. Brückner and Rutter [1] study the variant of the problem where the input comes with a fixed drawing of a subgraph and the question is whether the given drawing can be extended to a level-planar drawing of the whole graph without modifying the predrawn part. Their main result is that the problem can be solved in polynomial time if the input graph has a single source, and otherwise it is NP-complete. The hardness result holds under fairly strong restrictions, which include, e.g., a fixed embedding as well as bounded degree.

On the other hand, if we use the number s of sources in the input graph as our parameter, it is readily seen that there exists an XP-algorithm: Any level-planar drawing can be augmented to a level-planar drawing of a single-source graph by adding $s - 1$ edges. So we can simply guess beforehand $s - 1$ edges that we shall add to remove $s - 1$ sinks and then run the polynomial-time algorithm for single-source graphs. Our open question hence is, is the problem FPT with respect to the number of sources in the input graph?

References

- 1 Guido Brückner and Ignaz Rutter. Partial and constrained level planarity. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2000–2011. SIAM, 2017.
- 2 Guido Brückner, Ignaz Rutter, and Peter Stumpf. Level planarity: Transitivity vs. even crossings. In Therese C. Biedl and Andreas Kerren, editors, *Proceedings of the 26th International Symposium on Graph Drawing and Network Visualization (GD'18)*, volume 11282 of *Lecture Notes in Computer Science*, pages 39–52. Springer, 2018.
- 3 Radoslav Fulek, Michael J. Pelsmajer, Marcus Schaefer, and Daniel Štefankovič. Hanani-Tutte, Monotone Drawings, and Level-Planarity. In János Pach, editor, *Thirty Essays on Geometric Graph Theory*, pages 263–287. Springer New York, 2013.
- 4 Michael Jünger and Sebastian Leipert. Level planar embedding in linear time. *J. Graph Algorithms Appl.*, 6(1):67–113, 2002.
- 5 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.
- 6 Bert Randerath, Ewald Speckenmeyer, Endre Boros, Peter L. Hammer, Alexander Kogan, Kazuhisa Makino, Bruno Simeone, and Ondrej Čepek. A satisfiability formulation of problems on level graphs. *Electron. Notes Discret. Math.*, 9:269–277, 2001.

4.11 The Parameterized Complexity of Learning Small Decision Trees in Low-Dimensional Space

Manuel Sorge (TU Wien, AT)

License  Creative Commons BY 4.0 International license
© Manuel Sorge

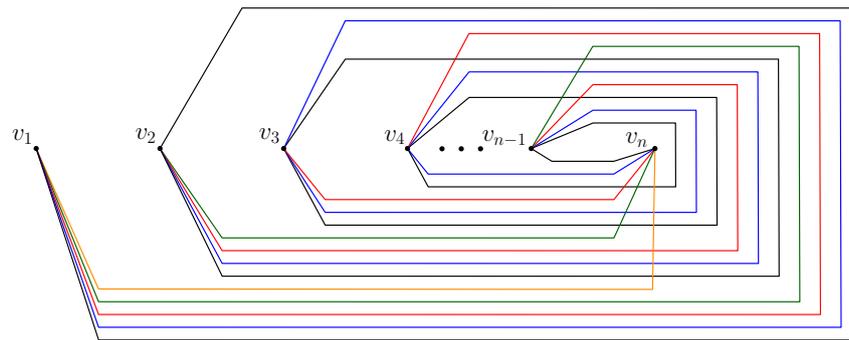
A basic machine-learning and data-analysis task is to classify a given set of examples together with their labels. That is, an *example set* is a set $E \subseteq \mathbb{R}^d$ together with a function $\lambda: E \rightarrow L$ for a label set L . A decision tree is a fundamental tool to classify example sets and can lead to particularly accessible and visual classifications. In a simple form, we have an ordered rooted tree, for each inner node a dimension from $\{1, 2, \dots, d\}$ and a threshold in \mathbb{R} , and each leaf of the tree has a label from L . To classify a given example e , we move through the tree as follows, starting from the root. At each node t we ask whether e 's entry in t 's dimension is less than or equal to the threshold at t . If so we move to the left child and otherwise to the right child. The class of the example e is then the label of the leaf of the tree at which we arrive in this manner. We say that the tree *decides* E if the class assigned by the tree to each example e in E agrees with $\lambda(e)$.

Heuristics for computing decision trees have been studied since at least the 1970s [1, 2] and many machine-learning libraries implement one of them. Apart from optimizing other parameters, often these heuristics minimize the size of the obtained decision tree. Hence the computational complexity of the following problem is interesting to know: In DECISION TREE SIZE we are given an example set E and want to compute the minimum size of a decision tree for E . DECISION TREE SIZE was known to be NP-complete since the 1970s [2]. However, to my knowledge, more fine-grained investigation into the complexity of DECISION TREE SIZE in form introduced above started only recently with [3]. In particular, the problem remains W[2]-hard with respect to the size of the tree and hence it is interesting to study the parameterized complexity of DECISION TREE SIZE with respect to other small parameters.

A mainstay in data analysis is performing dimensionality-reduction techniques, e.g. based on principal-component analysis, prior to using classification methods. The case where the number d of dimensions of the example space is small is thus an interesting special case. Marcin Pilipczuk pointed out to me that there is a simple dynamic programming algorithm that solves DECISION TREE SIZE in $n^{O(d)}$ time, where n is the number of input examples. In the seminar I asked: *Is DECISION TREE SIZE fixed-parameter tractable with respect to the number d of dimensions?*

References

- 1 L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Chapman & Hall/CRC, 1984. ISBN 0-534-98053-8.
- 2 L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, 5(1):15–17, 1976. doi: 10.1016/0020-0190(76)90095-8.
- 3 S. Ordyniak and S. Szeider. Parameterized complexity of small decision tree learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI '21)*, pages 6454–6462. AAAI Press, 2021.



■ **Figure 5** Construction taken from [3]: A simple drawing of K_n that does not contain a triangulation. (All empty triangles have either the edge v_1v_2 or the edge $v_{n-1}v_n$ on its boundary [3]. This drawing is also called the twisted drawing or Harborth's drawing.) The edges in this figure are colored for easier visibility; otherwise, the colors do not have any significance.

4.12 Two open problems on drawings of complete graphs

Birgit Vogtenhuber (TU Graz, AT)

License  Creative Commons BY 4.0 International license
© Birgit Vogtenhuber

Problem 1: Triangulations in simple drawings of the complete graph

A *simple drawing* of a graph $G = (V, E)$ in the plane is a drawing where vertices are distinct points, edges are Jordan arcs connecting their endpoints, and any pair of edges intersects at most once (either in a common endpoint or at a proper crossing in the relative interior of both edges). In a simple drawing D of a graph $G = (V, E)$, the edges of any three pairwise connected vertices form a Jordan curve that we call *triangle*. Any such triangle divides the plane into two regions. A triangle with vertices v_1, v_2, v_3 is called *empty* if the one of those regions does not contain any of the vertices $V \setminus \{v_1, v_2, v_3\}$.

A *triangulation* of a simple drawing D of the complete graph is a connected plane subdrawing of D in which every (bounded) face is an empty triangle (one might require the unbounded face of the subdrawing to be an empty triangle as well, or allow it to be a Jordan curve consisting of more than three edges of D). In simple drawings of the complete graph, any three vertices induce a triangle. However, not all simple drawings of the complete graph contain triangulations; see the below Figure for an example. This prompts the following open problem, which originally has been asked in [2].

Open Problem. What is the complexity of deciding whether a simple drawing of the complete graph contains a triangulation?

Related Results. Given a simple drawing D of the complete graph, and a cardinality k , it is NP-complete to decide whether there is a plane subdrawing of D that has at least k edges [7]. (This result has been proven in [7] via a reduction from the independent set problem: Given a set of segments in the plane that pairwise either are disjoint or intersect in a proper crossing, and an integer $k > 0$, it is NP-complete to decide whether there is a subset of k disjoint segments [4].) For straight-line drawings of the complete graph, it is easy to see that there always are triangulations. However, given a straight-line drawing of a non-complete graph, it is again NP-hard to decide whether it contains a triangulation [5].

Problem 2: The 2-colored crossing number for straight-line drawings of complete graphs

A *straight-line drawing* of G is a drawing D of G in the plane in which the vertices are drawn as points in general position, that is, no three points on a line, and the edges are drawn as straight line segments. A *2-edge-coloring* of D of a graph is an assignment of one of k possible colors to every edge of D . The *2-colored crossing number* of D is the minimum number of monochromatic crossings (pairs of edges of the same color that cross) in any 2-edge-coloring of D .

Open Problem. What is the complexity of deciding whether the 2-colored crossing number of a straight-line drawing D of the complete graph K_n is at most k ?

Remarks. Bounds on ratio between the 2-colored crossing number of a drawing D of K_n in relation to the total number of crossings in D have been studied in [1]. For (straight-line drawings of) general graphs, this problem is known to be NP-complete, even if the underlying point set in convex position [6]. The problem corresponds to finding a maximum cut in the segment intersection graph that is induced by the edges of the drawing.

References

- 1 Oswin Aichholzer, Ruy Fabila-Monroy, Adrian Fuchs, Carlos Hidalgo-Toscano, Irene Parada, Birgit Vogtenhuber, and Francisco Zaragoza. On the 2-colored crossing number. In Daniel Archambault and Csaba D. Tóth, editors, *Graph Drawing and Network Visualization*, pages 87–100, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-35802-0_7.
- 2 Oswin Aichholzer, Thomas Hackl, Alexander Pilz, Pedro Ramos, Vera Sacristán, and Birgit Vogtenhuber. Empty triangles in good drawings of the complete graph. *Graphs and Combinatorics*, 31(2):335–345, 2015. doi:10.1007/s00373-015-1550-5.
- 3 Heiko Harborth. Empty triangles in drawings of the complete graph. *Discrete Mathematics*, 191(1–3):109–111, 1998. doi:10.1016/S0012-365X(98)00098-3.
- 4 Jan Kratochví and Jaroslav Nešetřil. Independent set and clique problems in intersection-defined classes of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 31(1):85–93, 1990. URL: <http://eudml.org/doc/17810>.
- 5 Errol L. Lloyd. On triangulations of a set of points in the plane. In *Proc. 18th Annu. Symposium on Foundations of Computer Science*, pages 228–240, 1977. URL: <https://ieeexplore.ieee.org/document/4567947>.
- 6 S. Masuda, K. Nakajima, T. Kashiwabara, and T. Fujisawa. Crossing minimization in linear embeddings of graphs. *IEEE Transactions on Computers*, 39(1):124–127, 1990. doi:10.1109/12.46286.
- 7 Alfredo García Olaverri, Alexander Pilz, and Javier Tejel Altarriba. On plane subgraphs of complete topological drawings. *ARS MATHEMATICA CONTEMPORANEA*, 20(1):69–87, 2021. doi:10.26493/1855-3974.2226.e93.

5 Working groups

5.1 Progress on Upward Planarity Testing

Robert Ganian (TU Wien, AT), Steven Chaplick (Maastricht University, NL), Emilio Di Giacomo (University of Perugia, IT), Fabrizio Frati (University of Rome III, IT), Chrysanthi Raftopoulou (National Technical University of Athens, GR), and Kirill Simonov (TU Wien, AT)

License © Creative Commons BY 4.0 International license
 © Robert Ganian, Steven Chaplick, Emilio Di Giacomo, Fabrizio Frati, Chrysanthi Raftopoulou, and Kirill Simonov

The first problem considered by this working group is UPWARD PLANARITY TESTING (UP). In UP, 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.

UP 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 [8, 9], but the first polynomial-time algorithms for restricted variants of UP have been published already in the early nineties [13]. Among others, the problem is known to be polynomial-time tractable when G is provided with a plane embedding [1] (which also implies polynomial-time tractability for triconnected DAGs, since these admit a single plane embedding), or restricted to class of single-source DAGs [2], the class of outerplanar DAGs [16] or the class of orientations of series-parallel graphs [6].

In spite of the broad range of results for UP on specific subclasses of instances, the problem is considerably less explored from the parameterized complexity perspective. It was shown that UP is fixed-parameter tractable when parameterized by the cyclomatic number of the input DAG (or, equivalently, the feedback edge number of the underlying undirected graph of G) [5], and also when parameterized by the number of triconnected components and cut vertices [11].

We began our investigation by considering structural parameters for UP. Using standard reduction arguments, we could show that UP admits a polynomial kernel when parameterized by the vertex cover number of (the underlying undirected graph of) G . Moreover, we could strengthen these arguments to show that UP is fixed-parameter tractable when parameterized by the treedepth of (the underlying undirected graph of) G .

On a high level, the idea behind this result can be summarized as follows. We start by employing known results to compute a treedepth decomposition T for G [15]. T is a rooted tree over the vertices of G with the property that the endpoints of every arc in G have an ancestor-descendant relationship in T , with the property that the height of the tree (i.e., the maximum distance between a leaf and the root r) is at most the parameter value k . Let the level of a node v in T be its distance from the root r in T . Consider a node v on level i in T with the property that v has at least $f(k, i)$ -many children in T , for some well-defined and computable function f . We can then identify, in fixed-parameter time, a child w of v such that deleting the subtree of T rooted at w results in a subgraph which admits an upward planar drawing if and only if G admits an upward planar drawing. In other words, in this case we can reduce the size of the instance and restart the algorithm on a strictly smaller instance. On the other hand, if every node v on level i in T has at most $f(k, i)$ -many children, then G has size bounded by a function of k , and in particular it can be solved by a brute-force algorithm with runtime depending exclusively on k .

The general high-level approach outlined above is not entirely new, as it has been applied to obtain fixed-parameter algorithms for a handful of other problems parameterized by treedepth [7, 3]. However, the main technical challenge lies in the subtask of identifying a suitable child w that could be pruned from G if v had sufficiently many children, and in arguing that this operation is safe – i.e., that the resulting DAG is equivalent to G . To resolve this, we had to obtain a sufficient level of geometric insight into the problem’s behavior and apply a “swapping” argument whose details go beyond the scope of this brief report.

We then turned our attention to a different parameterization for the problem: the number of sources (s) and/or the number of sinks (t) of the input DAG. This was motivated by the fact that UP was shown to admit a non-trivial polynomial-time algorithm for the case when G has a single source [2]. Moreover, it is not difficult to observe that UP parameterized by s and/or by t can be reduced to the single-source case via branching that can be carried out in time $O(n^{\min(s,t)})$, which places the problem in XP for these parameterizations.

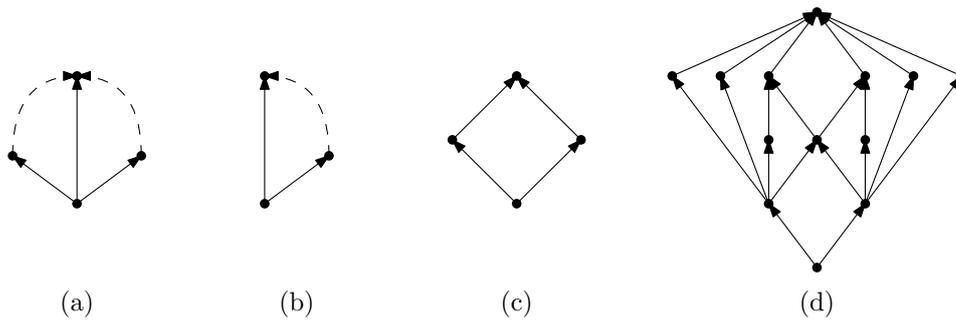
Our aim here was to determine whether this result could be strengthened to a fixed-parameter algorithm. While we made considerable progress towards this goal and are now convinced that this should be possible, some technical issues remain that we plan to address in follow-up virtual meetings. On the other hand, during our work we have already developed all the ingredients required to show that UP is fixed-parameter tractable when parameterized by $s + t$. The algorithm showing this is non-trivial, and a detailed summary exceeds the scope of this report: on a high level, we perform dynamic programming along the SPQR tree decomposition of G [12, 10], whereas we can show that at each node (which may be rigid, parallel or series, all of which must be handled separately) the number of decisions that need to be made and have an impact on whether the resulting drawing is upward planar or not can be upper-bounded to a function of k alone.

Last but not least, we briefly also considered the related problem of PARTIAL LEVEL PLANARITY TESTING (PLP). There, we are given an undirected graph G where each vertex is assigned to a level (i.e., an integer), and some subset H of the vertices are already drawn on the plane. The question is whether there exists a drawing of G which extends H and places each vertex of G in a way which matches the vertical levels prescribed on the input – in particular, two vertices must have the same y -coordinate if and only if they have the same level, vertex a has a higher y -coordinate than vertex b if and only if a has a higher level than b , and all edges monotonically increase in the vertical direction. Unlike UP, LEVEL PLANARITY TESTING (i.e., PLP when $H = \emptyset$) is polynomial-time tractable [14], but PLP is NP-hard in general [4]. As the final result for this report, we mention that we have made considerable progress towards showing that PLP is fixed-parameter tractable when parameterized by $|H|$; only a single technical hurdle remains, and we are optimistic that it will be resolved during the next follow-up meeting or two.

References

- 1 P. Bertolazzi, G. D. Battista, G. Liotta, and C. Mannino. Upward drawings of triconnected digraphs. *Algorithmica*, 12(6):476–497, 1994.
- 2 P. Bertolazzi, G. D. Battista, C. Mannino, and R. Tamassia. Optimal upward planarity testing of single-source digraphs. *SIAM J. Comput.*, 27(1):132–169, 1998.
- 3 S. Bhore, R. Ganian, F. Montecchiani, and M. Nöllenburg. Parameterized algorithms for queue layouts. In D. Auber and P. Valtr, editors, *Graph Drawing and Network Visualization – 28th International Symposium, GD 2020, Vancouver, BC, Canada, September 16-18, 2020, Revised Selected Papers*, volume 12590 of *Lecture Notes in Computer Science*, pages 40–54. Springer, 2020.

- 4 G. Brückner and I. Rutter. Partial and constrained level planarity. In P. N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2000–2011. SIAM, 2017.
- 5 H. Y. Chan. A parameterized algorithm for upward planarity testing. In S. Albers and T. Radzik, editors, *Algorithms – ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings*, volume 3221 of *Lecture Notes in Computer Science*, pages 157–168. Springer, 2004.
- 6 W. Didimo, F. Giordano, and G. Liotta. Upward spirality and upward planarity testing. *SIAM J. Discret. Math.*, 23(4):1842–1899, 2009.
- 7 R. Ganian and S. Ordyniak. The complexity landscape of decompositional parameters for ILP. *Artif. Intell.*, 257:61–71, 2018.
- 8 A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing, DIMACS International Workshop, GD '94, Princeton, New Jersey, USA, October 10-12, 1994, Proceedings*, volume 894 of *Lecture Notes in Computer Science*, pages 286–297. Springer, 1994.
- 9 A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM J. Comput.*, 31(2):601–625, 2001.
- 10 C. Gutwenger and P. Mutzel. A linear time implementation of spqr-trees. In J. Marks, editor, *Graph Drawing, 8th International Symposium, GD 2000, Colonial Williamsburg, VA, USA, September 20-23, 2000, Proceedings*, volume 1984 of *Lecture Notes in Computer Science*, pages 77–90. Springer, 2000.
- 11 P. Healy and K. Lynch. Two fixed-parameter tractable algorithms for testing upward planarity. *Int. J. Found. Comput. Sci.*, 17(5):1095–1114, 2006.
- 12 J. E. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM J. Comput.*, 2(3):135–158, 1973.
- 13 M. D. Hutton and A. Lubiw. Upward planar drawing of single source acyclic digraphs. In A. Aggarwal, editor, *Proceedings of the Second Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 28-30 January 1991, San Francisco, California, USA*, pages 203–211. ACM/SIAM, 1991.
- 14 M. Jünger, S. Leipert, and P. Mutzel. Level planarity testing in linear time. In S. 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.
- 15 J. Nešetřil and P. O. de Mendez. *Sparsity – Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012.
- 16 A. Papakostas. Upward planarity testing of outerplanar dags. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing, DIMACS International Workshop, GD '94, Princeton, New Jersey, USA, October 10-12, 1994, Proceedings*, volume 894 of *Lecture Notes in Computer Science*, pages 298–306. Springer, 1994.



■ **Figure 6** (a) The forbidden configuration, (b) a generalized triangle, (c) a rhombus, and (d) a graph that is not upward 2-page book embeddable.

5.2 Progress on Embedding Upward Planar Graphs in two Pages

Michael A. Bekos (Universität Tübingen, DE), Giordano Da Lozzo (University of Rome III, IT), Fabrizio Frati (University of Rome III, IT), Martin Gronemann (Universität Osnabrück, DE), and Chrysanthi Raftopoulou (National Technical University of Athens, GR)

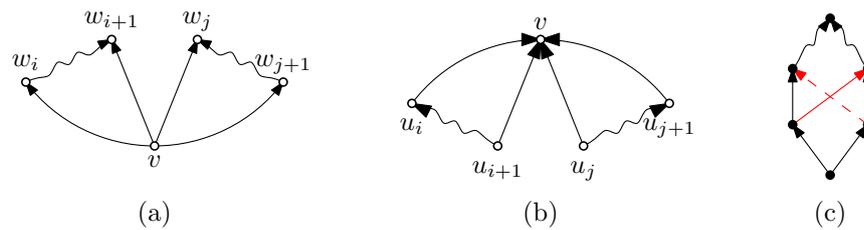
License © Creative Commons BY 4.0 International license

© Michael A. Bekos, Giordano Da Lozzo, Fabrizio Frati, Martin Gronemann, and Chrysanthi Raftopoulou

The general problem of whether an st -planar graph can be embedded into two pages or not has remained unanswered so far. However, for some special cases there exist efficient recognition algorithms. Mchedlidze and Symvonis [3] consider the case in which the input is triangulated. They show that if no transitive edge bounds two faces, the graph can be embedded in two pages. Furthermore, they prove that this condition is also sufficient for triangulations. Afterwards, Binucci et al. [1] extend this result to larger faces. Crucial in their work is the notion of forbidden configuration (see Fig. 6a, which consists of a transitive edge that bounds two internal faces (that are not necessarily triangles). Note that this is a generalization of the configuration used by Mchedlidze and Symvonis [3]. Clearly, the absence of forbidden configurations is a necessary condition for the existence of an upward 2-page book embedding. However, there exist examples that do not contain such a forbidden configuration and still do not admit an upward 2-page book embedding [4]; see Fig. 6d. Hence, the condition is not sufficient.

Based on their forbidden configuration Binucci et al. present in their work a linear-time recognition algorithm for a special class of st -planar graphs. Crucial in their paper is the notion of *generalized triangle*, i.e. an internal face bounded by a transitive edge; see Fig. 6b. Furthermore, they define a *rhombus* to be a face of size four that is not bounded by a transitive edge, i.e., the left and right path are both of length two; see Fig. 6c. They show that for an st -planar graph that solely consists of faces that are either a rhombus or a generalized triangle, one can decide in linear-time whether the graph admits an upward 2-page book embedding or not.

We tackle the problem from a different perspective and extend the concept of forbidden configurations. Central in our approach is the so-called bitonic st -ordering, which was introduced by Gronemann [2] to obtain upward planar polyline drawings of small size. Intuitively, a bitonic st -ordering forms a special type of st -ordering that takes the underlying embedding into account. The idea is that for a given embedding and an st -ordering, one considers the order of the successors of a vertex as they appear in the embedding.



■ **Figure 7** (a)-(b) The butterfly configuration, (c) splitting a large face.

Gronemann [2] showed that if these form an increasing and then decreasing sequence in the st -ordering, one may obtain an upward planar straight-line drawing of the underlying st -planar graph. We take this idea and adapt it to upward 2-page book embeddings.

Consider an st -planar graph $G = (V, E)$ and an upward 2-page book embedding of it. Clearly, the ordering of the vertices on the spine is a feasible st -ordering π for G . Moreover, the book embedding is a planar embedding of G . When now considering a vertex v and its successors $S(v) = \{w_1, \dots, w_s\}$ with $vw_i \in E$ and $1 \leq i \leq s$ ordered as they appear in the embedding, then one can observe that $\pi(w_1) > \dots > \pi(w_k) < \dots < \pi(w_s)$ holds for some $1 \leq k \leq s$. Symmetrically, we can make the same observations for all predecessors $P(v) = \{u_1, \dots, u_p\}$. That is $\pi(u_1) < \dots < \pi(u_l) > \dots > \pi(u_p)$ holds for some $1 \leq l \leq p$. In other words, the successors form a bitonic decreasing sequence w.r.t. to the spine ordering, while the predecessors form a bitonic increasing sequence. Gronemann [2] identified forbidden configurations in the graph that prevent the existence of st -orderings with such properties. We adapt these configurations to both the successors and predecessors. Figure 7 shows the forbidden configuration for both. We refer to these configurations as *butterfly*. Without proof, we observe:

► **Lemma 1.** *Let G be an embedded st -planar graph. If G contains a butterfly configuration, then G does not admit an upward 2-page book embedding.*

Note that a butterfly is a generalization of the forbidden configuration of Binucci et al. [1]. Our idea is to assume the absence of butterflies and augment the graph by adding edges. One promising strategy is to split large faces that are not generalized triangles to reduce the size of the largest face. In particular, we split such a large face into a smaller face and a rhombus; refer to Fig. 7c. The overall challenge with this approach is that one has two choices to perform such a split. While the augmentation with one edge is always possible, inserting the second edge may create too many restrictions.

References

- 1 C. Binucci, G. Da Lozzo, E. D. Giacomo, W. Didimo, T. Mchedlidze, and M. Patrignani. Upward book embeddings of st -graphs. In G. Barequet and Y. Wang, editors, *SoCG 2019*, volume 129 of *LIPICs*, pages 13:1–13:22. Schloss Dagstuhl, 2019.
- 2 M. Gronemann. Bitonic st -orderings for upward planar graphs. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization*, volume 9801 of *LNCS*, pages 222–235. Springer, 2016.
- 3 T. Mchedlidze and A. Symvonis. Crossing-optimal acyclic HP-completion for outerplanar st -digraphs. *JGAA*, 15(3):373–415, 2011.
- 4 R. Nowakowski and A. Parker. Ordered sets, pagenumbers and planarity. *Order*, 6(3):209–218, 1989.

5.3 Progress on A Parameterized Approach to Orthogonal Compaction

Philipp Kindermann (Universität Trier, DE), Walter Didimo (University of Perugia, IT), Siddharth Gupta (Ben-Gurion University, IL), Giuseppe Liotta (University of Perugia, IT), Alexander Wolff (Universität Würzburg, DE), and Meirav Zehavi (Ben-Gurion University, IL)

License  Creative Commons BY 4.0 International license

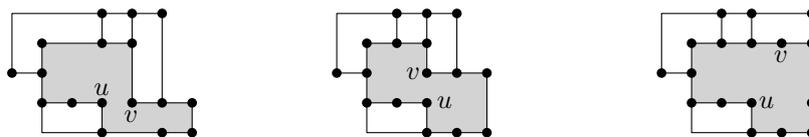
© Philipp Kindermann, Walter Didimo, Siddharth Gupta, Giuseppe Liotta, Alexander Wolff, and Meirav Zehavi

Background

Orthogonal Drawings and Representations. Let $G = (V, E)$ be a connected planar graph of vertex-degree at most four. A *planar orthogonal drawing* Γ of G maps each vertex $v \in V$ to a point p_v of the plane and each edge $e = (u, v) \in E$ to an alternating sequence of horizontal and vertical segments connecting p_u and p_v . A point of an edge of Γ in which a horizontal segment and a vertical segment meet is called a *bend*. We assume that in Γ all the vertices and bends have integer coordinates, i.e., we assume that Γ is an integer-coordinate grid drawing. Two planar orthogonal drawings Γ_1 and Γ_2 of G are *shape-equivalent* if: (i) Γ_1 and Γ_2 have the same planar embedding; (ii) for each vertex $v \in V$, the geometric angles at v (formed by any two consecutive edges incident on v) are the same in Γ_1 and Γ_2 ; (iii) for each edge $e = (u, v) \in E$ the sequence of left and right bends along e while moving from u to v is the same in Γ_1 and Γ_2 . An *orthogonal representation* (also called *orthogonal shape*) H of G is a class of shape-equivalent planar orthogonal drawings of G . It follows that an orthogonal representation H is completely described by a planar embedding of G , by the values of the angles around each vertex (each angle being a value in the set $\{90^\circ, 180^\circ, 270^\circ, 360^\circ\}$), and by the ordered sequence of left and right bends along each edge (u, v) , moving from u to v ; if we move from v to u this sequence and the direction (left/right) of each bend are reversed. If Γ is a planar orthogonal drawing in the class H , we also say that H is the orthogonal representation of Γ or that Γ *preserves* H . Without loss of generality, from now on we also assume that an orthogonal representation H comes with a given orientation, i.e., for each edge segment \overline{pq} of H (where p and q correspond to vertices or bends), we fix whether p lies to the left, to the right, above, or below q .

The Orthogonal Compaction Problem. Let H be an orthogonal representation of a connected planar graph G . The compaction problem for H asks to compute a minimum-area planar orthogonal drawing that preserves H . In other words, it asks to assign integer coordinates to the vertices and to the bends of H such that the area of the resulting planar orthogonal drawing is minimum among all planar orthogonal drawings that preserve H . In the following, we will refer to this problem as the **ORTHOGONAL COMPACTION (OC)** problem.

Previous Work. Patrignani proved that OC is NP-hard in the general case [5]; the problem remains NP-hard even for orthogonal representations of cycles [4]. Nevertheless, Bridgeman et al. [2] showed that OC can be solved in linear time for a subclass of orthogonal representations called *turn-regular*. Informally speaking, a face of a planar orthogonal representation H is turn-regular if it does not contain a pair of reflex corners (i.e., turns of 270°) that point to each other; H is turn-regular if all its faces are turn-regular.



■ **Figure 8** (Left) Drawing of a non-turn-regular orthogonal representation H ; vertices u and v point to each other in the gray face, thus they represent a pair of kitty corners. (Middle) Another drawing of H with minimum area. (Right) Drawing of a turn-regular orthogonal representation of the same graph.

More formally, let f be a face of H and assume that the boundary of f is always traversed counterclockwise (resp. clockwise) if f is internal (resp. external). Let c and c' be two reflex corners of f (corresponding to either vertices or bends)². Let $\text{rot}(c, c')$ be the number of convex corners minus the number of reflex corners encountered while traversing the boundary of f from c (included) to c' (excluded); a reflex corner corresponding to a vertex of degree one must be counted like two reflex corners. We say that c and c' is a pair of *kitty corners* of f if $\text{rot}(c, c') = 2$ or $\text{rot}(c', c) = 2$. A vertex is a kitty corner if it is part of a pair of kitty corners. Notice that the following property holds:

► **Property 1.** Let c and c' be two reflex corners of a face f . If f is internal, then $\text{rot}(c, c') = 2$ if and only if $\text{rot}(c', c) = 2$. If f is external, then $\text{rot}(c, c') = 2$ if and only if $\text{rot}(c', c) = -6$.

A face f of H is *turn-regular* if it does not contain a pair of kitty corners. The orthogonal representation H is *turn-regular* if all faces are turn-regular. Figure 8 shows two different drawings of the same orthogonal representation H that is not turn-regular, and a drawing of a turn-regular orthogonal representation of the same graph.

If H is turn-regular, then the compaction problem for H can be solved in linear time by independently solving two one-dimensional compaction problems for H ; one in the x -direction and the other in the y -direction [2]. Namely, for the x -direction, the one-dimensional compaction is solved as follows: (i) Construct a planar DAG D_x whose nodes are the maximal vertical chains of H and such that two nodes are connected by an arc (oriented from left to right) if the corresponding vertical chains are connected by a horizontal segment in H ; (ii) augment D_x to become a planar st-graph; (iii) apply an optimal topological numbering X to D_x (see [3], p. 89); the number $X(u)$ that is assigned to a node u determines the x -coordinate of all vertices of H in the vertical chain corresponding to u . The one-dimensional compaction in the y -direction is solved symmetrically.

Unfortunately, if H is not turn-regular, then the aforementioned approach fails; solving independently the one-dimensional compaction problem in the x - and in the y -directions may lead to non-planar drawings. This is due to the fact that, if c and c' form a pair of kitty corners, a directed path connecting the two (horizontal or vertical) maximal chains that include c and c' does not exist, neither in D_x nor in D_y .

Parameterized Analysis of the Compaction Problem

We initiate the study of the parameterized complexity of the OC problem. We study the complexity of the problem parameterized by the following parameters.

² For simplicity, one can assume that each bend of an orthogonal representation is replaced with a dummy vertex, so that all corners in a face correspond to vertices.

Kitty corners. Since the absence of kitty corners in an orthogonal representation suffices to solve OC efficiently, the most natural parameter to be considered is the number of kitty corners.

Number of Faces. Since OC remains NP-hard for orthogonal representations of cycles [4], we cannot expect an FPT (or even XP) algorithm parameterized by the number of faces of the embedded graph alone.

Face-Degree. The degree of a face is the number of vertices incident to the face, and the *face-degree* of an embedded graph is the maximum degree among all of its faces. Since both the NP-hardness reduction by Patrignani [5] and Evans et al. [4] require faces of linear size, it is interesting to know whether constant-size faces make the problem tractable.

Treewidth and Pathwidth. A *tree decomposition* of a graph $G = (V, E)$ is a tree $T = (V_T, E_T)$ and a function $\beta: V_T \rightarrow 2^V$ that assigns, to each node in T , a subset of vertices of G such that: (i) for each edge $(u, v) \in E$, the vertices u and v appear in a common subset; and (ii) for each vertex $v \in V$, the subsets in which v appears form a nonempty subtree (rather than just a subforest) in T . The *width* of T is one less than the size of the largest subset assigned by β , and the *treewidth* of G is the minimum width among all possible tree decompositions of G . Similarly, a *path decomposition* is a tree decomposition where T is a path, and the *pathwidth* of G is defined analogously to the treewidth of G .

Treewidth and pathwidth are among the most frequently used parameters in parameterized complexity. However, since cycles have constant pathwidth and treewidth (that is equal to 2), we also cannot expect an FPT (or even XP) algorithm parameterized by either of these two parameters alone.

Height. The *height* of a graph G is the minimum number of distinct y -coordinates required to draw the graph. In the case of orthogonal drawings, this is the same as the number of rows required. Since a $W \times H$ grid has pathwidth at most H , graphs with bounded height have bounded pathwidth, but the converse is generally not the case [1].

Our Results

We develop an XP algorithm, and then an FPT algorithm, parameterized by the number of kitty corners. For the XP algorithm, the idea is to “guess”, for each pair of kitty corners $\{u, v\}$, the relative positions of u and v , i.e., $x(u) \leq x(v)$ and $y(u) \leq y(v)$. For the FPT algorithm, more involved arguments are required to reduce the number of pairs of kitty corners for which the relative position has to be guessed. The rough idea is to explore a suitable set of planar edge augmentations of the two DAGs D_x and D_y resulting from the orthogonal representation H ; an augmenting edge connects a pair of nodes (i.e., vertical/horizontal chains of H) that involve a pair of kitty corners. For each combination of planar augmentations of D_x and D_y , one can further augment each of the two DAGs with edges that make it an *st*-planar graph, and then compute a pair of optimal topological numberings to determine the x - and the y -coordinates of a minimum-area drawing of H , within the given relative positions for the kitty corners.

► **Theorem 1.** *OC admits an XP algorithm parameterized by the number of kitty corners.*

► **Theorem 2.** *OC admits an FPT algorithm parameterized by the number of kitty corners.*

Note that the second theorem subsumes the first theorem. Unfortunately, our FPT algorithm does not imply a polynomial kernel for the problem. If we, however, take the sum of the number of kitty corners and the number of faces as parameter, then we can obtain a polynomial kernel.

► **Theorem 3.** *OC admits a polynomial kernel if parameterized by the sum of the number of kitty corners and the number of faces.*

By adjusting Patrignani’s NP-hardness proof for OC [5] accordingly, we show that the problem remains NP-hard even if all faces have constant degree.

► **Theorem 4.** *OC is NP-complete even for graphs of constant face-degree.*

By “guessing” for every column of the drawing which vertex or edge lies in each of the grid points, we obtain an XP algorithm parameterized by the height of the graph.

► **Theorem 5.** *OC admits an XP algorithm parameterized by the height of the graph.*

So, while OC is unlikely to admit an XP algorithm with respect to pathwidth, it admits an XP algorithm with respect to height. This motivates us to define a parameterization that lies “in-between” pathwidth and height: that can be arbitrarily smaller than height, yet yield an XP algorithm. In addition to OC, we prove that our new parameterization is of independent interest, being relevant to several other problems in Graph Drawing.

Future Work

The following questions remain open, and will be part of our future research.

- Can we find a polynomial kernel for OC with respect to only the number of kitty corners?
- Does OC admit an FPT algorithm parameterized by the height of the graph?
- Is OC solvable in $2^{O(\sqrt{n})}$ time? This bound is tight assuming that the Exponential Time Hypothesis is true.

References

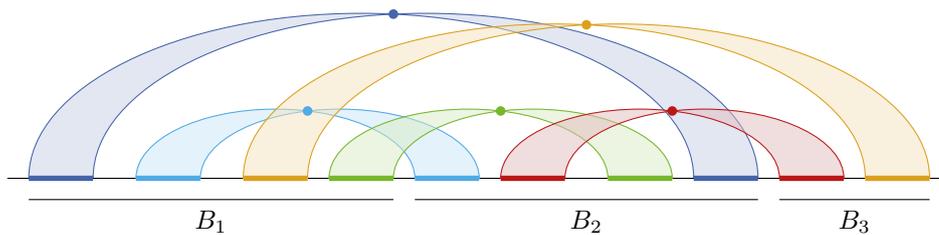
- 1 Therese C. Biedl. Small drawings of outerplanar graphs, series-parallel graphs, and other planar graphs. *Discret. Comput. Geom.*, 45(1):141–160, 2011. DOI: 10.1007/s00454-010-9310-z
- 2 Stina S. Bridgeman, Giuseppe Di Battista, Walter Didimo, Giuseppe Liotta, Roberto Tamassia, and Luca Vismara. Turn-regularity and optimal area drawings of orthogonal representations. *Comput. Geom.*, 16(1):53–93, 2000. DOI: 10.1016/S0925-7721(99)00054-1
- 3 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, Upper Saddle River, 1999.
- 4 William S. Evans, Krzysztof Fleszar, Philipp Kindermann, Noushin Saeedi, Chan-Su Shin, and Alexander Wolff. Minimum rectilinear polygons for given angle sequences. *Comput. Geom.*, 100(101820):1–39, 2022. DOI: 10.1016/j.comgeo.2021.101820
- 5 Maurizio Patrignani. On the complexity of orthogonal compaction. *Comput. Geom.*, 19(1):47–67, 2001. DOI: 10.1016/S0925-7721(01)00010-4

5.4 Progress on Almost Separated Fixed Order Stack Layouts

Johannes Zink (Universität Würzburg, DE), Martin Gronemann (Universität Osnabrück, DE), Thekla Hamm (TU Wien, AT), Boris Klemz (Universität Würzburg, DE), Martin Nöllenburg (TU Wien, AT), and Birgit Vogtenhuber (TU Graz, AT)

License © Creative Commons BY 4.0 International license
 © Johannes Zink, Martin Gronemann, Thekla Hamm, Boris Klemz, Martin Nöllenburg, and Birgit Vogtenhuber

In the course of the workshop, we jointly developed the following preliminary results on the open problem on *Almost Separated Fixed Order Stack Layouts*. We briefly recall the setting and fix some terminology: As instance we consider a graph G together with an ordering σ



■ **Figure 9** Instance with $k = 3$ and largest twist size t that needs at least $3t/2$ stacks.

of its vertices and $s \in \mathbb{N}$. Additionally we define the *number of blocks* of σ as the smallest number k such that $V(G)$ can be partitioned into k sets B_1, \dots, B_k of vertices which we call *blocks* that are consecutive in σ such that for all i , and two vertices $v, v' \in B_i$, v and v' are not connected by an edge in G . It is straightforward to observe that G must be p -colorable for some $p \leq k$. Our task is to decide whether the fixed-order stack number of G with respect to σ , denoted by $\text{fosn}(G, n)$, is at most s , using k as a parameter.

A natural lower bound for the fixed-order stack number is the size t of a largest *twist* of a graph, i.e., a maximum size set of mutually intersecting edges (note that the order of the vertices determines which edges intersect when drawn on the same page). An interesting structural observation is that, even for constant k , t plus an additive constant does not upper-bound the fixed-order stack number s as we show by the following simple observation.

► **Proposition 1.** For $k = 3$, there are bipartite graphs and vertex orderings requiring $3t/2$ stacks, where t is the size of the largest twist.

Proof sketch. Each colored bundle in Fig. 9 represents a twist of size $t/2$ and one can easily verify that the largest twist of the entire instance has size t . Since the red and green bundles intersect, they need their separate sets of $t/2$ pages each. The dark blue bundle can be added to pages with green edges, while the light blue bundle can be added to pages with red edges (otherwise they would already increase the fixed-order stack number). Since the orange bundle, however, intersects both blue bundles, it has to use a new set of $t/2$ pages, resulting in a fixed-order stack number of $3t/2$. ◀

On the positive side, we provide the following XP algorithm and approximation algorithm.

► **Theorem 1.** Deciding whether a graph $G = (V, E)$ with respect to a given ordering σ of V with k blocks has fixed-order stack number $\leq s$, is in XP in $s + k$. More precisely, there is an algorithm deciding whether $\text{fosn}(G, \sigma) \leq s$ in time $O(m^{4sk+5/2} 4^{sk} sk)$, where $m = |E|$.

Proof sketch. On each page of a fixed-order stack layout with respect to σ , the set of edges between any two blocks must pairwise nest, i.e. no two edges have endpoints that alternate in σ . We call them a *rainbow*. Our goal is to find rainbows of different block pairs that we can place onto the same page. Here, a key observation is that we will only need to know the top and the bottom edge of the rainbow to decide whether rainbows cross.

We branch on the set of all top and bottom edges of all rainbows for all block pairs and all pages in a hypothetical solution. The number of rainbows on any page of a hypothetical fixed-order stack layout is at most $2k$; this is because for any page contracting each rainbow on the page into an edge and then contracting each block into a vertex yields an outerplanar graph on at most k vertices and hence with $\leq 2k - 3$ edges. Hence overall we have $O(m^{4ks})$ branches. Note that we consider also all hypothetical solutions that use less than $(2k - 3)s$ rainbows since we may select edges multiple times, which corresponds to selecting a smaller set of top or bottom edges.

Our next step is to associate pairs of top and bottom edges for each rainbow and nest the remaining edges into appropriate rainbows. In each branch, we construct the *nesting digraph* N , whose vertices correspond to the edges of G and there is an arc between vertices corresponding to edges e and e' if e' nests inside e . We remove all incoming arcs from vertices that represent top edges and all outgoing arcs from vertices that represent bottom edges. By definition, N is acyclic which allows us to find a minimal path cover in time $O(|V(N)|^{5/2}) = O(m^{5/2})$. Each path in the resulting path cover will correspond to a rainbow. For correctness it is important to note that in this path cover, paths may connect vertices associated to top and bottom edges of different rainbows in a fixed hypothetical solution associated to the current branch. However, we can argue that such a hypothetical solution can be transformed into one that has the same pairs of top and bottom rainbow edges that are connected via our path cover, by a careful exchange argument.

Finally, we assign the rainbows to pages in a way in which they do not cross. For this we consider the *conflict graph* where the rainbows obtained in the previous step correspond to vertices, and edges connect vertices corresponding to rainbows that are of the same block pair or cross. These are precisely the rainbows that may not be placed onto the same page. A proper vertex coloring of the conflict graph with at most s colors immediately corresponds to a page assignment of the rainbows and hence yields a fixed-order stack layout on at most s pages for the computed set of rainbows. Since the conflict graph has $2sk$ vertices, we can find an s -coloring in time $O(2^{2sk}sk)$, or decide that it does not exist.

Correctness follows from our earlier ‘key observation’ and the fact that if there is a solution at some point we can assume to consider the same pairs of top and bottom edges for all rainbows. ◀

► **Theorem 2.** *There is an $O(m^{5/2})$ -time $(k - 1)$ -approximation algorithm for determining the fixed-order stack number of a graph $G = (V, E)$ with respect to a given ordering σ of V , where $m = |E|$ and k is the number of blocks of σ .*

Proof sketch. Construct the (directed acyclic) compatibility graph C as follows. Add a vertex for each edge of G and add an arc if two edges can be placed onto the same page. Formally, for $\sigma = (v_1, v_2, \dots, v_n)$, $e_i = v_{i_1}v_{i_2}$ and $e_j = v_{j_1}v_{j_2}$, there is an arc from e_i to e_j if $i_1 \leq j_1 < j_2 \leq i_2$ (i.e., e_j nests inside e_i) or if $i_1 < i_2 \leq j_1 < j_2$ (i.e., e_i and e_j form a *necklace*). Find a minimum path cover in C in $O(m^{5/2})$ time. Wherever a path uses a necklace arc, split the path into two. For each resulting path, use its own page.

Clearly, each resulting path is a rainbow and, thus, its edges can be placed onto the same page. Any path in C uses $\leq k - 1$ necklace arcs. Since our initial path cover used $\leq \text{fosn}(G, \sigma)$ paths (this follows from the fact that an optimal solution can be represented by $\text{fosn}(G, \sigma)$ paths in C), we have $\leq (k - 1) \text{fosn}(G, \sigma)$ paths/rainbows after splitting. ◀

5.5 Progress on Applications of the Product Structure Theorems

Giordano Da Lozzo (University of Rome III, IT), Michael A. Bekos (Universität Tübingen, DE), Petr Hlinený (Masaryk University – Brno, CZ), and Michael Kaufmann (Universität Tübingen, DE)

License © Creative Commons BY 4.0 International license
© Giordano Da Lozzo, Michael A. Bekos, Petr Hlinený, and Michael Kaufmann

Consider two graphs A and B . The *strong product* of A and B , denoted by $A \boxtimes B$, is the graph such that: (i) $V(A \boxtimes B) = V(A) \times V(B)$ and (ii) there exists an edge between the vertices $(a_1, b_1), (a_2, b_2) \in V(A \boxtimes B)$ if and only if one of the following occurs: (a) $a_1 = a_2$

and $b_1b_2 \in E(B)$, (b) $b_1 = b_2$ and $a_1a_2 \in E(A)$, or $a_1a_2 \in E(A)$ and $b_1b_2 \in E(B)$. Our research group considered strong products that yield supergraphs of k -planar graphs, which are defined as follows. A graph is k -planar if it admits a k -planar drawing, i.e., a drawing in the plane in which each edge is crossed at most k times. We exploit these products to derive new upper bounds on the queue number of k -planar graphs. Recall that, the *queue number* $qn(G)$ of a graph G corresponds to the minimum size of the largest rainbow over all linear orderings of the vertices of G , where a *rainbow* is a set of independent nested edges.

In a recent preprint [3], Dujmović, Morin, and Wood have proved the following theorem on the structure of k -planar graphs.

► **Theorem 1.** *Every k -planar graph is a subgraph of the strong product of three graphs $H \boxtimes P \boxtimes K_{18k^2+48k+30}$, where H is a planar graph of treewidth at most $\binom{k+4}{3} - 1$ and P is a path.*

Furthermore, for the special case in which $k = 1$, the same authors proved the following stronger statement.

► **Theorem 2.** *Every 1-planar graph is a subgraph of the strong product of three graphs $H \boxtimes P \boxtimes K_{30}$, where H is a planar graph of treewidth at most 3 and P is a path.*

In [2], Dujmović et al. have proved the following useful lemma concerning the queue number of graphs that can be expressed as (subgraphs of) the strong product of three graphs exhibiting the properties of Theorems 1 and 2.

► **Lemma 3.** *If $G \subseteq H \boxtimes P \boxtimes K_\ell$ then $qn(G) \leq 3\ell qn(H) + \lfloor \frac{3}{2}\ell \rfloor$.*

Combining Lemma 3 and Theorem 1, Dujmović, Morin, and Wood showed the first constant upper bound on the queue number of k -planar graphs [3], thus resolving a long-standing open question. Furthermore, applying Lemma 3 and Theorem 2, they improved this bound to 495 for 1-planar graphs. We observe, in particular, that every improvement to the ‘ K_{30} ’ term of Theorem 2, would immediately improve the bound on the queue number of 1-planar graphs, as well as other related results.

In this working group, we researched in the direction of improving Theorem 2 for 1-planar graphs. We also investigated a possible generalization of these ideas to k -planar graphs for $k \geq 2$. In this regard, we considered the family of optimal 2-planar graphs, and more in general the larger graph family of h -framed graphs: An h -framed graph is a connected graph that admits a drawing in the plane such that the removal of all its crossed edges yields a bi-connected plane graph with faces of size at most h . This graph family was first introduced by Bekos et al. [1], in the context of the counterpart of queue layouts, called *stack layouts*. We remark that the family of h -framed graphs strictly contains the ones of triconnected 1-planar and optimal 2-planar graphs, as the graphs in these families can be augmented to 4-framed and 5-framed graphs, respectively [1].

We state below, without proving, our main claims.

- By carefully redesigning some of the arguments from [3], namely the choices that determine the value ℓ of the graph K_ℓ in Lemma 3, we believe it is possible to improve Theorem 2 to the product $H \boxtimes P \boxtimes K_7$. By Lemma 3, this would immediately improve the upper bound on the queue number of 1-planar graphs to 115.
- Essentially the same improved approach leads to a result that any h -framed graph is a subgraph of the strong product of three graphs $H \boxtimes P \boxtimes K_{O(h)}$. In particular, an upper bound on the size of the complete graph involved in the product seems to be $\lfloor \frac{5h}{2} \rfloor - 3$;

although, several details have to be worked out. Therefore, for h -framed graphs, we obtain the first non-trivial upper bound on the queue number of the graphs in this family that only depends on h .

- Further investigations led to new ideas which could probably improve Theorem 1 for 2-planar graphs to a product $H \boxtimes P \boxtimes K_{33}$. However, this is the subject of ongoing research.
- We provide a new strong product theorem for 1-planar graphs of the form $H \boxtimes P^2 \boxtimes K_3$, where H is a planar graph of treewidth at most 3 and P^2 is the square of a path P . This, in turn, leads to a further improvement of the queue number of 1-planar graphs.

All these informal claims are yet to be written down with proper proofs and verified (at the time of writing up this report).

References

- 1 M. A. Bekos, G. Da Lozzo, S. Griesbach, M. Gronemann, F. Montecchiani, C. N. Raftopoulou: Book Embeddings of Nonplanar Graphs with Small Faces in Few Pages. *SoCG 2020*: 16:1-16:17
- 2 V. Dujmović, G. Joret, P. Micek, P. Morin: Planar Graphs Have Bounded Queue-Number. *J. ACM*,67(4): 22:1-22:38 (2020).
- 3 V. Dujmović, P. Morin, and D. Wood: Graph product structure for non-minor-closed classes. *CoRR* abs/1907.05168 (2020).

5.6 Progress on the Parameterized Complexity of Small Decision Tree Learning

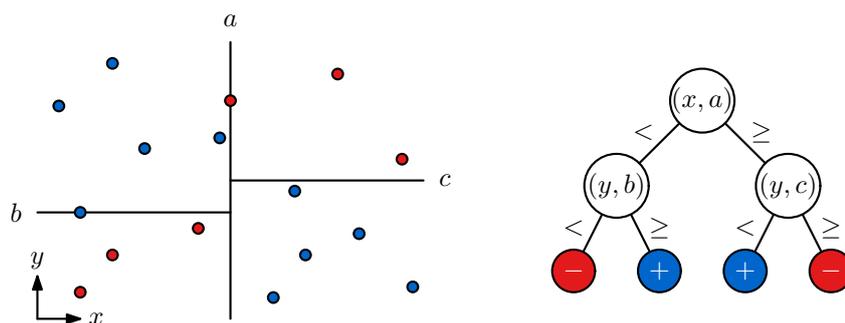
Stephen G. Kobourov (University of Arizona – Tucson, US), Maarten Löffler (Utrecht University, NL), Fabrizio Montecchiani (University of Perugia, IT), Raimund Seidel, Ignaz Rutter (Universität Passau, DE), Manuel Sorge (TU Wien, AT), and Jules Wulms (TU Wien, AT)

License © Creative Commons BY 4.0 International license
 © Stephen G. Kobourov, Maarten Löffler, Fabrizio Montecchiani, Raimund Seidel, Ignaz Rutter, Manuel Sorge, and Jules Wulms

Decision Trees are well known tools used to describe, classify, and generalize data. Besides their simplicity, decision trees are particularly attractive for providing interpretable models of the underlying data.

The setting is as follows (see Figure 10). The task is to classify an *example set*, a set $E \subseteq \mathbb{R}^d$ together with a function $\lambda: E \rightarrow \{\oplus, \ominus\}$ labeling each example with a *class*. For this task, a *decision tree* is a rooted binary tree T together with two functions $\dim: V(T) \rightarrow [d]$ and $\text{thr}: V(T) \rightarrow \mathbb{R}$ that label each inner node $t \in V(T)$ by a *dimension* $\dim(t) \in [d]$ and a *threshold* $\text{thr}(t) \in \mathbb{R}$. For each inner node t of T the edges to the two children of t are labeled by *yes* and *no*. Each node $t \in V(T)$, including the leaves, defines a subset $E[T, t] \subseteq E$ as follows: Consider the path P from the root of T to t . An example $e \in E$ is in $E[T, t]$ if and only if, for each node v on P , it holds that $e[\dim(v)] \leq \text{thr}(v)$ if the edge following v on P is *yes* and it holds $e[\dim(v)] > \text{thr}(v)$ if the edge following v on P is *no*. If the tree T is clear from the context, we simplify $E[T, t]$ to $E[t]$.

A decision tree T is a *decision tree for* (E, λ) if for each leaf ℓ of T we have that all examples in $E[\ell]$ have the same label under λ . The *size* of a decision tree is the number of its inner nodes.



■ **Figure 10** An instance (E, λ) of DTS with a minimum decision tree T for (E, λ) . Examples labeled \oplus and \ominus by λ are blue and red respectively. Each internal node $t \in T$ is labeled by $(\dim(t), \text{thr}(t))$.

In this working group we studied the complexity of the problem MINIMUM DECISION TREE SIZE (DTS). The input consists of an integer s and an example set (E, λ) . The task is to decide whether there exists a decision tree for (E, λ) that has size at most s . Our aim was to solve the open question about the parameterized complexity of DTS with respect to the number d of dimensions of the example space, posed by Manuel Sorge in the same seminar.

We did not feel comfortable to immediately and directly attack the open question and first explored the behavior of the problem in particular with respect to other well-motivated parameters. Besides the number d of features (or dimensions) and the natural size-parameter s of the decision tree, additional interesting parameters are the maximum number of features (or dimensions) on which any two examples differ δ_{max} , and the maximum number of different values a feature ranges over (the domain size) D_{max} . Ordyniak and Szeider [1] proved that DTS parameterized by s is $W[2]$ -hard already when each feature is binary, and hence DTS is $W[2]$ -hard also when parameterized by $s + D_{max}$. Moreover, the same reduction shows that the problem is **paraNP**-hard when parameterized by $\delta_{max} + D_{max}$. On the positive side, DTS parameterized by s lies in **XP**. The main positive result in Ref. [1] establishes an **FPT** algorithm for DTS parameterized by $s + \delta_{max} + D_{max}$. It is open whether the problem is **FPT** parameterized by $s + \delta_{max}$.

We first observed that some small adaptations of an algorithm of Ordyniak and Szeider [1] shows that DTS is fixed-parameter tractable when parameterized by $s + d$. On the other hand, the hardness result in Ref. [1] shows that the DTS is **paraNP**-hard also when parameterized by the minimum number of examples in one of the two classes (called r). Indeed, the reduction is such that one of the two classes contains only one example. However, the number d of dimensions in the reduction is unbounded. A natural question is therefore whether DTS parameterized by $d + r$ is **FPT**. We answered this question in the positive, and generalized the result to a more general parameterization, namely $d + R$, where R is the minimum number of leaves of the same class.

We explored various directions regarding the relation of r and δ_{max} , the development of efficient data-reduction rules, and the structure of the hypergraph defined by sets of dimensions in which pairs of examples differ. These directions did not bear tangible results, except for counterexamples for intuitive statements such as “if an example is directly between examples of the same class in all dimensions, then it can be safely removed”. Exploring these directions did however provide us with much better intuition about the behavior of the problem. Ultimately, this led to a construction that with high confidence can be used

to show that DTS is $W[1]$ -hard with respect to the number d of dimensions. This solves the open question and complements the fact that DTS can be solved in $O(D_{max}^{2d+1}d)$ time (personal communication with Marcin Pilipczuk).

References

- 1 S. Ordyniak and S. Szeider. Parameterized complexity of small decision tree learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI '21)*, pages 6454–6462. AAAI Press, 2021.

Participants

- Michael A. Bekos
Universität Tübingen, DE
- Steven Chaplick
Maastricht University, NL
- Giordano Da Lozzo
University of Rome III, IT
- Emilio Di Giacomo
University of Perugia, IT
- Walter Didimo
University of Perugia, IT
- Fabrizio Frati
University of Rome III, IT
- Robert Ganian
TU Wien, AT
- Martin Gronemann
Universität Osnabrück, DE
- Thekla Hamm
TU Wien, AT
- Petr Hlinený
Masaryk University – Brno, CZ
- Michael Kaufmann
Universität Tübingen, DE
- Philipp Kindermann
Universität Trier, DE
- Boris Klemz
Universität Würzburg, DE
- Stephen G. Kobourov
University of Arizona –
Tucson, US
- Giuseppe Liotta
University of Perugia, IT
- Maarten Löffler
Utrecht University, NL
- Fabrizio Montecchiani
University of Perugia, IT
- Martin Nöllenburg
TU Wien, AT
- Chrysanthi Raftopoulou
National Technical University of
Athens, GR
- Ignaz Rutter
Universität Passau, DE
- Kirill Simonov
TU Wien, AT
- Manuel Sorge
TU Wien, AT
- Birgit Vogtenhuber
TU Graz, AT
- Alexander Wolff
Universität Würzburg, DE
- Jules Wulms
TU Wien, AT
- Johannes Zink
Universität Würzburg, DE

Remote Participants

- Meirav Zehavi
Ben-Gurion University –
Beer Sheva, IL
- Siddharth Gupta
Ben-Gurion University –
Beer Sheva, IL

