Leibniz Universität Hannover

Institut für Theoretische Informatik

72. Theorietag

(17. & 18. November 2016)

Gesellschaft für Informatik

TheorieTag

Program and Workshop Information

Contents

1 Program on Thursday

Thursday, 17th November 2016	
12:00	Lunch
13:30	**Malte Skambat (Kiel)**: Offline Drawing of Dynamic Trees: Algorithmics and Document Integration
14:00	**Philipp Zschoche / Leon Kellerhals (Berlin)**: On the Computational Complexity of Variants of Combinatorial Voter Control in Elections
14:30	**Nils Vortmeier (Dortmund)**: Dynamic Complexity under Definable Changes
15:00	Coffee break
15:30	**Rolf Niedermeier (Berlin):** Parameterized Algorithmics–On Interactions with Heuristics
16:45	Members' meeting: Fachgruppe ALGO
17:15	Members' meeting: Fachgruppe KP
19:00	Dinner at Himalaya Restaurant (Postkamp 18)

2 Program on Friday

Friday, 18th November 2016	
9:30	**Anna-Sophie Himmel (Berlin)**: Enumerating Maximal Cliques in Temporal Graphs
10:00	**Till Fluschnik (Berlin)**: Fractals for Kernelization Lower Bounds, With an Application to Length-Bounded Cut Problems
10:30	Coffee break
11:00	**Martin Lück (Hannover)**: The Complexity of Computation Tree Logic
11:30	**Florentin Neumann (Koblenz/Hamburg)**: An Asynchronous Distributed Algorithm for Finding Hamiltonian Cycles in Random Graphs
12:00	**Maurice Chandoo (Hannover)**: On the Implicit Graph Conjecture
12:30	Closing and Lunch

3 Overview of Talks

Parameterized Algorithmics—On Interactions with Heuristics

Rolf Niedermeier (TU Berlin)

Parameterized algorithm design is mainly tailored towards identifying "tractable" special cases for "intractable" (that is, typically NP-hard) problems. Ideally, this leads to efficient algorithms providing optimal solutions. The central observation herein is that if some problem-specific parameters are small, then certain problems can be solved efficiently by confining exponential running time growth to the parameters only.
In real-world scenarios, most computationally hard problems are attacked with heuristic approaches, that is, often simple (in particular, greedy) algorithms that are efficient but do not guarantee optimal solutions, or algorithms without provable running time guarantees.
As Richard Karp pointed out, a long-term goal of Theoretical Computer Science is to contribute to a better understanding of the effectiveness of heuristics.
In this talk, through some case studies including examples from graph-based data clustering, graph anonymization, and computational social choice, we discuss some fruitful interactions between heuristics and parameterized algorithm design and analysis.

Offline Drawing of Dynamic Trees: Algorithmics and Document Integration

Malte Skambath (Universität Kiel)

Main Reference Malte Skambath, Till Tantau.
Offline Drawing of Dynamic Trees: Algorithmics and Document Integration.
To appear in the Proceedings of the 24th International Symposium on Graph Drawing and Network Visualization (GD 2016).

While the algorithmic drawing of static trees is well-understood and well-supported by software tools, creating animations depicting how a tree changes over time is currently difficult: software support, if available at all, is not integrated into a document production workflow and algorithmic approaches only rarely take temporal information into consideration. During the production of a presentation or a paper, most users will visualize how, say, a search tree evolves over time by manually drawing a sequence of trees. We present an extension of the popular TEX typesetting system that allows users to specify dynamic trees inside their documents, together with a new algorithm for drawing them. Running TEX on

the documents then results in documents in the SVG format with visually pleasing embedded animations. Our algorithm produces animations that satisfy a set of natural aesthetic criteria when possible. On the negative side, we show that one cannot always satisfy all criteria simultaneously and that minimizing their violations is NP-complete.

On the Computational Complexity of Variants of Combinatorial Voter Control in Elections

Philipp Zschoche (TU Berlin)

Joint work of Leon Kellerhals, Viatcheslav Korenwein, Philipp Zschoche, Robert Bredereck, and Jiehua Chen

Voter control problems model situations in which an external agent tries to affect the result of an election by adding or deleting the fewest number of voters. The goal of the agent is to make a specific candidate either win (constructive control) or lose (destructive control) the election. We study the constructive and destructive voter control problems when adding and deleting voters have a combinatorial flavor, meaning that if we add or delete a voter v, we also add or delete a bundle $\kappa(v)$ of voters that are associated with v. We analyze the computational complexity of the four voter control problems for the Plurality rule. We obtain that, in general, making a candidate lose is computationally easier than making her win. In particular, if the bundling relation is symmetric (i.e. $\forall w : w \in \kappa(v) \Leftrightarrow v \in \kappa(w)$), and if each voter has at most two voters associated with him, then destructive control by deleting the fewest number of voters or candidates are polynomial-time solvable while the constructive variants remain NP-hard. Even if the bundling relation consists of disjoint cliques (i.e. $\forall w : w \in \kappa(v) \Leftrightarrow \kappa(v) = \kappa(w)$), the constructive problem variants remain intractable. Finally, constructive control by adding the fewest number of voters does not admit an efficient approximation algorithm, unless $\mathsf{P} = \mathsf{NP}$.

Dynamic Complexity under Definable Changes

Nils Vortmeier (TU Dortmund)

Joint work of Nils Vortmeier, Thomas Schwentick, and Thomas Zeume

A dynamic program, as introduced by Dong, Su and Topor and Patnaik and Immerman, maintains the result of a fixed query for an input database which is subject to changes. It can use

an auxiliary database whose relations are updated via first-order formulas upon changes of the input database.
In the original setting, only insertions and deletions of single tuples are considered as changes of the database. In this talk we will allow changes defined by (restricted) first-order formulas and review which queries can still be maintained.
Many maintenance results for single-tuple changes can be extended to more powerful change operations: for example Reachability for undirected graphs is first-order maintainable under single-tuple changes and first-order defined insertions, likewise Reachability for directed acyclic graphs under quantifier-free insertions. On the other hand, several inexpressibility results are obtained, for example that the reachability query cannot be maintained by quantifier-free programs under definable, quantifier-free changes.

Enumerating Maximal Cliques in Temporal Graphs

Anne-Sophie Himmel (TU Berlin)

Main Reference Anne-Sophie Himmel, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. *Enumerating Maximal Cliques in Temporal Graphs.* Preprint available in CoRR abs/1605.03871.

Dynamics of interactions play an increasingly important role in the analysis of complex networks. A modeling framework to capture this are temporal graphs. We focus on enumerating Delta-cliques, an extension of the concept of cliques to temporal graphs: for a given time period Delta, a Delta-clique in a temporal graph is a set of vertices and a time interval such that all vertices interact with each other at least after every Delta-time steps within the time interval. Viard, Latapy, and Magnien [1] proposed a greedy algorithm for enumerating all maximal Delta-cliques in temporal graphs. In contrast to this approach, we adapt to the temporal setting the Bron-Kerbosch algorithm—an efficient, recursive backtracking algorithm which enumerates all maximal cliques in static graphs. We obtain encouraging results both in theory (concerning worst-case time analysis based on the parameter "Delta-slice degeneracy" of the underlying graph) as well as in practice with experiments on real-world data. The latter culminates in a significant improvement for most interesting Delta-values concerning running time in comparison with the algorithm of Viard, Latapy, and Magnien (typically two orders of magnitude).

References

[1] Jordan Viard, Matthieu Latapy, and Clémence Magnien. "Revealing contact patterns among high-school students using maximal cliques in link streams". In: *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2015*. 2015, pp. 1517–1522.

Fractals for Kernelization Lower Bounds, With an Application to Length-Bounded Cut Problems

Till Fluschnik (TU Berlin)

Main Reference Till Fluschnik, Danny Hermelin, André Nichterlein, Rolf Niedermeier.
Fractals for Kernelization Lower Bounds, With an Application to Length-Bounded Cut Problems.
ICALP 2016: 25:1-25:14.

The composition technique is a popular method for excluding polynomial-size problem kernels for NP-hard parameterized problems. We present a new technique exploiting triangle-based fractal structures for extending the range of applicability of compositions. Our technique makes it possible to prove new no-polynomial-kernel results for a number of problems dealing with length-bounded cuts. In particular, answering an open question of Golovach and Thilikos [1], we show that, unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$, the NP-hard Length-Bounded Edge-Cut (LBEC) problem (delete at most k edges such that the resulting graph has no s-t path of length shorter than ℓ) parameterized by the combination of k and ℓ has no polynomial-size problem kernel. Our framework applies to planar as well as directed variants of the basic problems and also applies to both edge and vertex deletion problems.

References

[1] Petr A. Golovach and Dimitrios M. Thilikos. "Paths of bounded length and their cuts: Parameterized complexity and algorithms". In: *Discrete Optimization* 8.1 (2011), pp. 72–86.

The Complexity of Computation Tree Logic

Martin Lück (Leibniz Universität Hannover)

Main Reference Martin Lück. *Quirky Quantifiers: Optimal Models and Complexity of Computation Tree Logic.*
Preprint available in CoRR abs/1510.08786.

The satisfiability problem of the branching time logic CTL is studied in terms of computational complexity. A sharp dichotomy is shown in terms of complexity and minimal models: Temporal depth one has low expressive power, while temporal depth two is equivalent to full CTL.

An Asynchronous Distributed Algorithm for Finding Hamiltonian Cycles in Random Graphs

Florentin Neumann (Hamburg University of Technology)

Joint work of Florentin Neumann and Volker Turau

Random graphs are a commonly used tool to depict and analyze statistical behavior of large networks [1]. In the classical $G(n,p)$ model of random graphs by Erdős and Rényi [2], a graph $G \sim G(n,p)$ is a finite undirected random graph with n nodes where each edge independently exists with probability p.

There is a large body of work (e.g. [3, 4, 5, 6, 7]) on algorithms for computation of Hamiltonian cycles in random graphs which succeed with high probability (w.h.p.). A Hamiltonian cycle is a cycle that contains every node exactly once. An algorithm is said to succeed w.h.p., if it outputs the desired result with probability converging to 1 as n goes to infinity.

It is well known, that the decision problem, whether or not a graph contains a Hamiltonian cycle, is NP-complete. The aforementioned algorithms rely on the fact [8, Th. 8.9] that a graph $G \sim G(n,p)$ contains w.h.p. a Hamiltonian cycle, while $p = (\log n + \log\log n + \omega(n))/n$ and where $\omega(n)$ is a sequence with $\lim_{n\to\infty} \omega(n) = \infty$. All of these algorithms, however, are sequential algorithms. In fact, the only distributed message passing algorithm for computing Hamiltonian cycles in random graphs is the algorithm by Levy et al. [9]. Executed on a graph $G \sim G(n,p)$, their algorithm outputs w.h.p. a Hamiltonian cycle given that $p = \omega(\sqrt{\log n}/n^{1/4})$. This algorithm works in synchronous distributed systems (i.e., nodes are assumed to operate in synchronous rounds), terminates in linear worst-case number of rounds, and requires $O(n^{3/4+\epsilon})$ rounds on expectation. It is assumed that a dedicated *root* node is fixed and that all nodes know the total number of nodes n as well as the identifiers of their direct neighbors.

We complement this line of research by presenting (ongoing) research on *asynchronous distributed algorithms* for computation of Hamiltonian cycles. Executed on $G \sim G(n,p)$ with $p = c\log n/n$ and $c > 1$, the presented algorithm succeeds w.h.p. after at most $O(n^2/\mathrm{polylog}(n))$ rounds. The size of a message is $O(\log n)$ bits and nodes require only $O(n)$ bits local space. The algorithm proceeds in three phases: In the first phase, a depth first search tree is constructed starting from any fixed root node, using standard distributed techniques. W.h.p., the longest paths in this tree consists of all but $O(n(\log\log n)/\log n)$ nodes [10, Th. 6.6]. In the second phase, the remaining nodes are successively integrated

into the longest path using a standard technique called rotation-extension. The proof of existence of a Hamiltonian path requires execution of an unlimited number of rotation-extensions. This concept cannot be translated directly into an algorithm. We overcome this difficulty by allowing only specific sequences of rotation-extensions, called forward rotation-extensions. These can be generated with complexity $O(n)$. We prove that if the path is not extendable by means of such a forward rotation-extension, then w.h.p. the path is already a Hamiltonian path. The third phase converts w.h.p. this Hamiltonian path into a Hamiltonian cycle, using the same technique.
Compared to the previous work by Levy et al. [9], the algorithm presented here succeeds w.h.p. for much smaller p and under relaxed (asynchronous) system assumptions.

References

[1] Krzysztof Krzywdzinski and Katarzyna Rybarczyk. "Distributed algorithms for random graphs". In: *Theor. Comput. Sci.* 605 (2015), pp. 95–105.

[2] Paul Erdős and Alfréd Rényi. "On Random Graphs I." In: *Publicationes Mathematicae (Debrecen)* 6 (1959 1959), pp. 290–297.

[3] Lajos Pósa. "Hamiltonian circuits in random graphs". In: *Discrete Mathematics* 14.4 (1976), pp. 359–364. issn: 0012-365X.

[4] Dana Angluin and Leslie G. Valiant. "Fast Probabilistic Algorithms for Hamiltonian Circuits and Matchings". In: *J. Comput. Syst. Sci.* 18.2 (1979), pp. 155–193.

[5] Eli Shamir. "How many random edges make a graph Hamiltonian?" In: *Combinatorica* 3.1 (1983), pp. 123–131.

[6] Béla Bollobás, Trevor Ian Fenner, and Alan Frieze. "An algorithm for finding hamilton paths and cycles in random graphs". In: *Combinatorica* 7.4 (1987), pp. 327–341. issn: 1439-6912.

[7] Andrew Thomason. "A simple linear expected time algorithm for finding a hamilton path". In: *Discrete Mathematics* 75.1-3 (1989), pp. 373–379.

[8] Béla Bollobás. *Random Graphs*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2001. isbn: 9780521797221.

[9] Eythan Levy, Guy Louchard, and Jordi Petit. "A Distributed Algorithm to Find Hamiltonian Cycles in Random Graphs". In: *Combinatorial and Algorithmic Aspects of Networking, First Workshop on Combinatorial and Algorithmic Aspects of Networking, CAAN 2004, Banff, Alberta, Canada, August 5-7, 2004, Revised Selected Papers*. 2004, pp. 63–74.

[10] Alan Frieze and Michał Karoński. *Introduction to Random Graphs*. Introduction to Random Graphs. Cambridge University Press, 2015. isbn: 9781107118508.

On the Implicit Graph Conjecture

Maurice Chandoo (Leibniz Universität Hannover)

Main Reference Maurice Chandoo. *On the Implicit Graph Conjecture.* Proc. 41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland, pp. 23:1–23:13, http://dx.doi.org/10.4230/LIPIcs.MFCS.2016.23

The implicit graph conjecture states that every sufficiently small, hereditary graph class has a labeling scheme with a polynomial-time computable label decoder. We approach this conjecture by investigating classes of label decoders defined in terms of complexity classes such as P and EXP. For instance, GP denotes the class of graph classes that have a labeling scheme with a polynomial-time computable label decoder. Until now it was not even known whether GP is a strict subset of GR. We show that this is indeed the case and reveal a strict hierarchy akin to classical complexity. We also show that classes such as GP can be characterized in terms of graph parameters. This could mean that certain algorithmic problems are feasible on every graph class in GP. Lastly, we define a more restrictive class of label decoders using first-order logic that already contains many natural graph classes such as forests and interval graphs. We give an alternative characterization of this class in terms of directed acyclic graphs. By showing that some small, hereditary graph class cannot be expressed with such label decoders a weaker form of the implicit graph conjecture could be disproven.

4 Workshop Information

Venue

The workshop is located at the Institute for Theoretical Computer Science, at

Appelstraße 4, 30167 Hannover

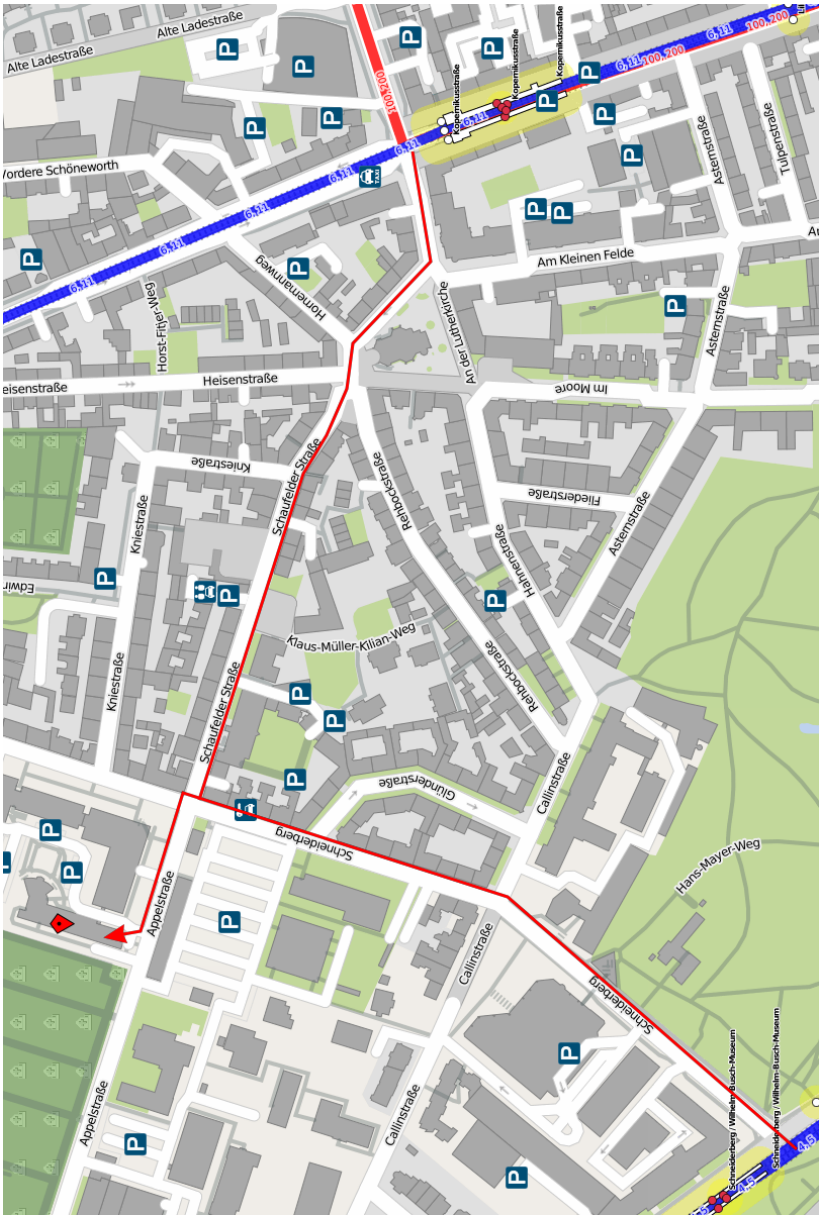
and will take place in room 224 on the second floor. The building can be reached by car, or from the city center by metro lines 4-Garbsen/5-Stöcken until Schneiderberg, or 6-Nordhafen/11-Haltenhoffstraße until Kopernikusstraße. After you entered the building use the stairs up to the second floor. There, just use the door bell left of the glass door. The door is programmed to automatically open after ringing the bell during the workshop time. The workshop is located at the end of the floor in the last room on the right.

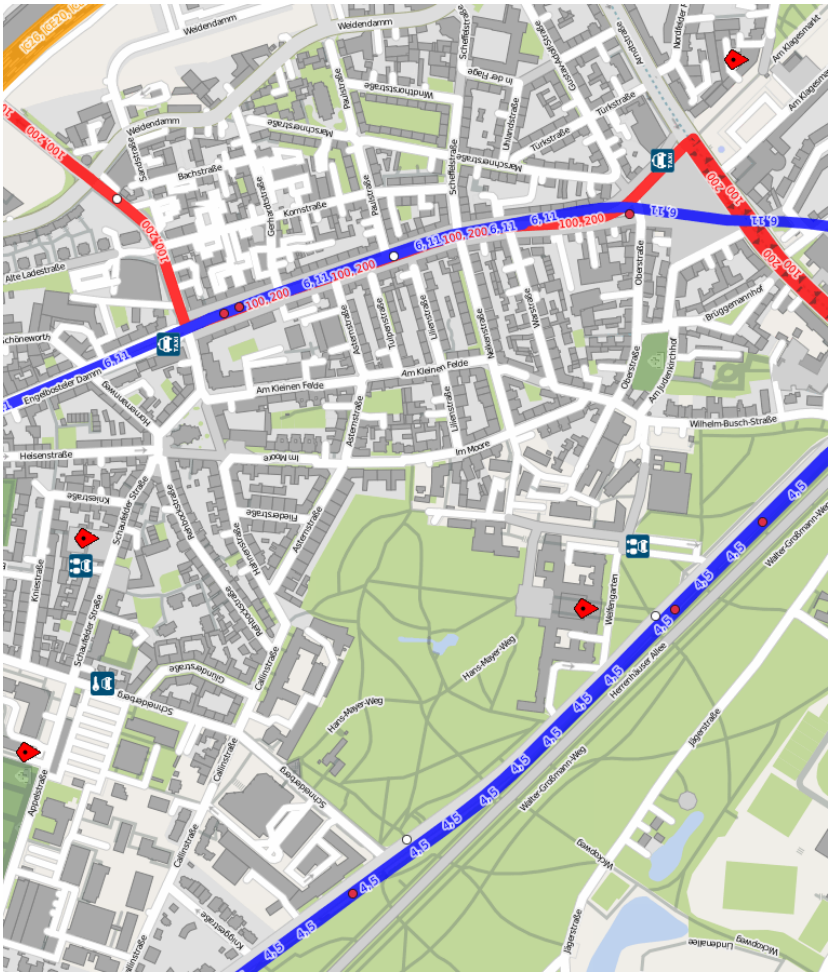
Wifi Access

The *eduroam* wireless network, and a guest network will be available.

Lunch and Dinner

On Thursday, lunch will take place at *Kaiser* (http://www.gaststaette-kaiser.de) and at Friday at *Zwischenzeit* (www.restaurant-zwischenzeit.de). On Thursday, the workshop dinner will take place at the Himalaya restaurant (www.himalaya-hannover.de) at Postkamp 18.
The participants are invited to join the organizers on the way to the lunch location (approximately at 11:50 on Thursday, resp., after the end of the workshop on Friday) and/or to the dinner location (approximately at 18:20 on Thursday).
We will meet outside at the main entrance of the computer science building (Appelstraße 4).





Marked (left to right):

- ⋆ Workshop location (Appelstraße 4)
- ⋆ Zwischenzeit restaurant (Schaufelder Straße 11)
- ⋆ Leibniz Universität main building (Welfengarten 1)
- ⋆ Himalaya restaurant (Am Klagesmarkt/Postkamp 18)