Dagstuhl Seminar 10441 Final Report

Exact Complexity of NP-Hard Problems

31 October – 5 November 2010

Table of Contents

1 Organization 1

2 Executive summary 2
 2.1 Background 2
 2.2 The meeting 3

3 Participants and group photo 5

4 Abstracts of presentations 7

5 Open problems 15

1 Organization

The seminar was organized by:

Thore Husfeldt, Lund University, SE and ITU Copenhagen, DK, Thore.Husfeldt@cs.lth.se
Dieter Kratsch, Université Paul Verlaine–Metz, FR, kratsch@univ-metz.fr
Ramamohan Paturi, University of California, San Diego, US, paturi@cs.ucsd.edu
Gregory Sorkin, London School of Economics, London, GB, G.B.Sorkin@lse.ac.uk

Abstracts and open problems were compiled and edited by
Serge Gaspers, Vienna University of Technology, Vienna, AT, gaspers@kr.tuwien.ac.at, who also assisted in preparation of this final report.

We are grateful to the Dagstuhl personnel for their helpfulness and expertise, making the meeting smooth-running, pleasurable, productive, and easy to organize.

2 Executive summary

2.1 Background

A decade before NP-completeness became the lens through which Computer Science views computationally hard problems, beautiful algorithms were discovered that are much better than exhaustive search, for example Bellman's 1962 dynamic programming treatment of the Traveling Salesman problem and Ryser's 1963 inclusion–exclusion formula for the permanent.

Today we know that all NP-hard problems are unlikely to admit polynomial-time algorithms, yet NP-hard problems *must* be solved, in some way, for everything from manufacturing and supply-chain optimization to railroad timetabling. Approaches include approximation algorithms, heuristics, average-case analysis, and exact exponential-time algorithms: all are essential. While all NP-complete problems are equivalent from the polynomial-time perspective, their exponential-time properties vary widely. Which problems are easiest or hardest? What are the promising algorithmic techniques? What are the connections with parametrized complexity? How fast an algorithm can we find? What about complexity lower bounds?

Work addressing such questions, both from the algorithmic and complexity theoretic sides, has become known as *exact complexity*. Despite significant progress, the area is still fairly new and many fundamental problems remain open. Where the approximation algorithms field, for example, has unifying algorithmic techniques such as LP rounding and semidefinite programming, and hardness techniques from probabilistically checkable proofs and the Unique Games conjecture, much exact algorithms work is still specific to a particular NP-complete problem: powerful unified techniques are just emerging.

Exciting new results and directions have been established by scientists on several continents, with important contributions coming from young researchers such as Williams and Traxler. The purpose of this workshop is to accelerate developments in this late-blooming field. Below, we outline several new results and promising directions.

The Tutte polynomial of an n-vertex, m-edged graph can trivially be evaluated in time $O^*(2^m)$, but no vertex-parameterized algorithm is obvious. The Potts (q-coloring) partition function can trivially be evaluated in time $O^*(q^n)$, but it is not obvious if one can remove the dependence on q. The Fortuin–Kasteleyn model from statistical physics generalizes both, and a breakthrough result of Björklund, Husfeldt, Kaski, and Koivisto [FOCS 2006, STOC 2007, FOCS 2008] shows how to evaluate it using the *inclusion–exclusion method* in time $O^*(2^n)$. It is an intriguing question as to how far these techniques could be extended.

Recently, the color-coding technique of Alon, Yuster, and Zwick [JACM 1995] has been extended by introducing algebraic structures that yield faster fixed parameter tractable algorithms. Koutis [ICALP 2008] uses "vector coding" for a randomized $O^*(2^{3k/2})$ algorithm for the k-Path problem, and Williams [IPL 2009] improves this to $O^*(2^k)$. Such *algorithms from group algebra* are a promising direction for further exploration.

Branch-and-reduce is one of the most frequently used methods for solving NP-hard problems, but current analyses of such algorithms may be overly pessimistic. Fomin, Grandoni and Kratsch [ICALP 2005, SODA 2006] used a *measure and conquer* framework to establish simple and fast algorithms to solve the Minimum Dominating Set and the Maximum Independent Set problem. By now measure and conquer analysis has had an enormous impact. This and related methods, Eppstein's quasiconvex analysis [SODA 2004], Scott and Sorkin's linear programming method [Random 2005], and Gaspers and Sorkin's convex programming method [SODA 2009], have become indispensable, but a need remains for further improvements.

Faster algorithms, notably for Maximum Independent Set, have resulted from *computer-produced* graphical reductions and case analysis. Can these automated techniques be put on a more general theoretical level, and improved? Can similar automation be applied to *logic-based branching rules* such as the "clause learning" of Kulikov and Kutzkov [CSR 2007]? What about lower bounds on such local methods?

Exponential-time and other approaches may be combined. Scott and Sorkin's [CPC 2006] *average-case analysis* of an exponential-time algorithm shows that Max 2-CSP is solvable in expected linear time on random constraint graphs below the giant-component threshold. It would be interesting to obtain similar results for other problems. Cygan, Kowalik, Pilipczuk and Wykurz [2008] explore *exponential-time approximation algorithms* trading off the running time and approximation ratio for various problems, an approach well worth further investigation. *Hybrid algorithms*, introduced by Vassilevska, Williams and Woo [SODA 2006], are exemplified for Bandwidth: the fastest exact algorithm known takes time $O^*(5^n)$ ($O^*(4.383^n)$ as of April 2010) and the best polynomial-time approximation ratio is roughly $O(\log^3 n)$, but there is a hybrid algorithm that, depending on the input, produces either the exact minimum in time roughly $O^*(4^n)$ or an $O(\log^{2.5} n)$ approximation in polynomial time.

Some other approaches merit mention. Horowitz and Sahni's $O^*(2^{n/2})$ Knapsack algorithm [JACM 1974] is an early example of the approach of reducing a hard problem to one that is *easy but exponentially large.* Williams' $O^*(1.74^n)$ algorithm for Max 2-CSP [ICALP 2004] is the only nontrivial algorithm known for *dense instances.* Koivisto's ring extension of this algorithm [IPL 2006], the Tutte and k-Path work previously mentioned, and Scott and Sorkin's ring extension of other Max 2-CSP algorithms [TALG 2009] show the power of *algebraic approaches.*

Despite these algorithmic successes, there may be limits to what is possible. Impagliazzo, Paturi and Zane [FOCS 1998] initiate a complexity theory of exact algorithms, using subexponential-time Turing reductions. They introduce the Exponential-Time Hypothesis (ETH), viz., that the best exact algorithm for 3-SAT has exponential complexity 2^{cn} where $c > 0$. Assuming ETH they prove that the best exponent sequence c_k for k-SAT must be an increasing sequence [CCC 1999]. Using similar techniques, Traxler [IWPEC 2008] shows that for 2-CSPs over k-valued variables, assuming ETH, any algorithm's running time must depend strongly on k; this is in sharp contrast to the earlier-mentioned result that k-coloring can be solved in time $O^*(2^n)$, independent of k. These results call for a vigorous investigation of the complexity-theoretic limitations of exact algorithms.

2.2 The meeting

The meeting was attended by 46 researchers, the maximum possible modulo some last-minute cancellations. The organizers are grateful to all who came, and regret that — due to a gratifyingly high acceptance rate — others who would have contributed could not be invited. The participants came from around the globe, predominantly from Europe as usual for this field, but this time also with a good showing from the US.

AT	CZ	DE	DK	FI	FR	GB	IL	IN	IT	JP	NL	NO	PL	RU	SE	US
1	2	7	2	2	4	3	1	1	1	2	3	3	1	2	2	9

Comparing with the Seminar 08431 on Moderately Exponential Time Algorithms held in October 2008, along with exponential-time algorithms this meeting had stronger representation in fixed-parameter tractability and in computational complexity, both of which furthered a sense of the field cementing into a cohesive and substantial discipline. Paturi opened the technical proceedings with a survey of the complexity background, in particular various exponential time hypotheses. Dell introduced an exponential time hypothesis for counting problems and Lokshtanov introduced another hypothesis about the complexity of algorithms for Satisfiabilty.

Structurally, most talks were half an hour long, with exceptions made for a handful of hour-long talks, and participant feedback during and after the meeting indicates that this was a good choice. The after-lunch period was left free for informal discussions and small working groups, with talks again between 4pm and dinnertime, and participants liked this as well. Rather than having a single, massive, open problem session, the meeting included half-hour long open problem sessions right before dinner on every day except Wednesday. These sessions brought out many interesting yet approachable problems (see the Open Problems section of this report), a tribute to the state of the field or the openness of the participants. Wednesday included the traditional outing but with a plethora of options: participants could hike, run, or bike (all in the rain; that was not optional), and could go on afterward to a wine tasting and dinner or enjoy a quiet evening at the Schloss.

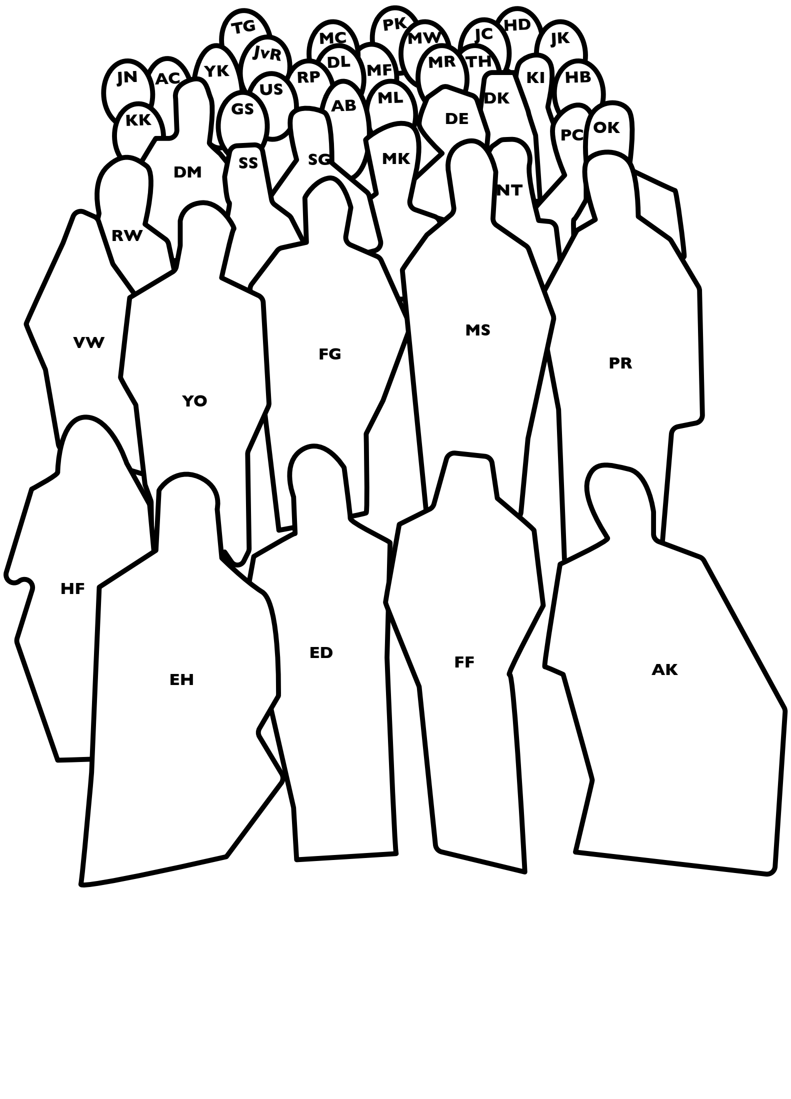
Talks included a wide range of concrete results on NP-hard problems, for example a nearly optimal algorithm for listing all maximal cliques in a graph (Eppstein, Löffler, Strash) and a proof of fixed-parameter tractability (FPT) of the multi-way cut problem (Marx, Razgon), drawing upon and extending techniques such as those discussed in the Background section above. One result that stood out (Lokshtanov, Marx, Saurabh) was that known treewidth-parametrized algorithms for Independent Set, Dominating Set, Max Cut, q-Coloring, Odd Cycle Transversal, and Partition Into Triangles are all essentially optimal, assuming a hypothesis about the complexity of Satisfiability; the proof is based on small-treewidth gadget-based reductions of boolean formulas to these problems. A complete presentation was given of the ideas in the recent breakthrough on the Hamilton Cycle problem (Björklund). For an n-vertex graph, Hamilton Cycle has a naive algorithm running in time $n!$, and can be solved in time $O^*(2^n)$ using dynamic programming (Bellman 1962, Held and Karp 1962), but has resisted further improvement. Björklund showed an elegant algorithm for bipartite graphs running in time $O^*(2^{n/2})$, and how to turn it into a general algorithm running in time $O^*(1.657^n)$. The algorithm takes determinants over an appropriate algebra, building on Koutis' and Williams' algebraic sieving methods for k-path.

Several open problems from previous Dagstuhl seminars, notably Seminar 08431, were resolved. Johan van Rooij had asked for a Capacitated Dominating Set algorithm faster than 2^n, and Mathieu Liedloff presented one with running time $O^*(1.8463^n)$. Mikko Koivisto had asked a related, broad question about reducibility among problems for which the best known algorithms take time 2^n: if one can be solved faster, can others? While not explicitly answering this question, the results above by Lokshtanov, Saurabh, and Marx bear on it by showing that 2^n is probably optimal for some of these problems. In Woeginger's famous 2003 survey of exact algorithms, the first open problem was to find algorithms faster than 2^n for Hamilton cycle and the traveling salesman problem; Björklund's algorithm satisfies the first challenge.

Seminar participants can visit the meeting's wiki at http://tinyurl.com/355og64 and Thore Husfeldt has blog entries on the meeting's subject and the process of organizing it at http://tinyurl.com/2vbxdx6 and http://tinyurl.com/3912axz.

3 Participants and group photo

Andreas Björklund (Lund University)
Hans L. Bodlaender (Utrecht University)
Paolo Codenotti (University of Chicago)
Amin Coja-Oghlan (University of Warwick)
Jean-Francois Couturier (Université Paul Verlaine - Metz)
Marek Cygan (University of Warsaw)
Evgeny Dantsin (Roosevelt University - Chicago)
Holger Dell (HU Berlin)
David Eppstein (University of California - Irvine)
Henning Fernau (Universität Trier)
Fedor V. Fomin (University of Bergen)
Martin Fürer (Pennsylvania State University)
Serge Gaspers (TU Wien)
Tomas Gavenciak (Charles University - Prague)
Fabrizio Grandoni (Università di Roma II)
Edward A. Hirsch (Steklov Institute - St. Petersburg)
Thore Husfeldt (ITU and Lund University)
Kazuo Iwama (Kyoto University)
Petteri Kaski (Aalto University)
Mikko Koivisto (University of Helsinki)
Yiannis Koutis (Carnegie Mellon University - Pittsburgh)
Jan Kratochvil (Charles University - Prague)
Dieter Kratsch (Université Paul Verlaine - Metz)
Alexander S. Kulikov (Steklov Institute - St. Petersburg)
Oliver Kullmann (University of Wales - Swansea)
Konstantin Kutzkov (IT University of Copenhagen)
Mathieu Liedloff (Université d'Orleans)
Daniel Lokshtanov (University of California - San Diego)
Dániel Marx (HU Berlin)
Daniel Meister (Universität Trier)
Jesper Nederlof (University of Bergen)
Yoshio Okamoto (JAIST)
Ramamohan Paturi (UC San Diego)
Mike Robson (Université Bordeaux)
Peter Rossmanith (RWTH Aachen)
Michael Saks (Rutgers University - Piscataway)
Saket Saurabh (The Institute of Mathematical Sciences - Chennai)
Uwe Schöning (Universität Ulm)
Gregory B. Sorkin (London School of Economics)
Ewald Speckenmeyer (Universität Köln)
Nina Sofia Taslaman (IT University of Copenhagen)
Johan van Rooij (Utrecht University)
Magnus Wahlström (MPI für Informatik - Saarbrücken)
Ryan Williams (IBM Almaden Center - San José)
Virginia Vassilevska Williams (University of California - Berkeley)
Gerhard Woeginger (TU Eindhoven)


TG
PK
HD
MC
MW
JC
JK
JvR
DL
MR
TH
YK
MF
KI
HB
JN
AC
RP
US
ML
DK
GS
AB
DE
KK
PC
OK
SS
SG
MK
DM
NT
RW
VW
FG
MS
PR
YO
HF
ED
FF
AK
EH

4 Abstracts of presentations

4.1 Determinant Sums (and Labeled Walks) for Undirected Hamiltonicity

Andreas Björklund (Lund University, SE)

We present a Monte Carlo algorithm for Hamiltonicity detection in an n-vertex undirected graph running in $O^*(1.657^n)$ time. To the best of our knowledge, this is the first super-polynomial improvement on the worst case runtime for the problem since the $O^*(2^n)$ bound established for TSP almost fifty years ago (Bellman 1962, Held and Karp 1962). It answers in part the first open problem in Woeginger's 2003 survey on exact algorithms for NP-hard problems.

For bipartite graphs, we improve the bound to $O^*(1.414^n)$ time. Both the bipartite and the general algorithm can be implemented to use space polynomial in n.

We combine several recently resurrected ideas to get the results. Our main technical contribution is a new reduction inspired by the algebraic sieving method for k-Path (Koutis ICALP 2008, Williams IPL 2009). We introduce the Labeled Cycle Cover Sum in which we are set to count weighted arc labeled cycle covers over a finite field of characteristic two. We reduce Hamiltonicity to Labeled Cycle Cover Sum and apply the determinant summation technique for Exact Set Covers (Björklund STACS 2010) to evaluate it.

We also present an alternative algorithm by B., Husfeldt, Kaski, and Koivisto 2010, based on labeled walks which we feel is even simpler. We will show how both techniques can be altered to solve for the parametrized k-Path problem in $\mathsf{poly}(n)1.657^k$ time.

Keywords: Hamiltonian Cycle

Full Paper: http://arxiv.org/abs/1008.0541

See also: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science

4.2 An exact algorithm for Intervalizing k-colored graphs

Hans L. Bodlaender (Utrecht University, NL)

The Intervalizing k-colored graphs problem has as input a properly vertex-colored graph G, and asks if G is subgraph of a properly colored interval graph. This problem is known to be NP-complete, even if the number of colors is four. Here, we present an exact algorithm for the problem, with a somewhat curious running time, which is sub-exponential: it uses $O(2^{n/log^{1-\epsilon}})$ time, for all $\epsilon > 0$. The algorithm is a simple modification on a Held-Karp-style algorithm, incorporating an isomorphism test on certain subgraphs.

Keywords: Interval graphs, subexponential time, interval completion

4.3 Isomorphism of hypergraphs of low rank in sub-exponential time

Paolo Codenotti (University of Chicago, US)

We give an algorithm to decide isomorphism of hypergraphs of rank k in time $\exp(\tilde{O}(k^2\sqrt{n}))$, where n is the number of vertices. (The rank is the maximum size of edges; the tilde refers to a polylogarithmic factor.) The case of bounded k answers a 24-year-old question and removes an obstacle to improving the worst case-bound for Graph Isomorphism testing. The best previously known bound, even for $k = 3$, was C^n (Luks 1999).

Keywords: Hypergraph Isomorphism, Groups, Sub-exponential time

Joint work of: Babai, Laszlo; Codenotti, Paolo;

See also: László Babai and Paolo Codenotti, Isomorphism of hypergraphs of low rank in moderately exponential time, FOCS 2008.

4.4 On belief propagation guided decimation for random k-SAT

Amin Coja-Oghlan (University of Warwick, GB)

Random k-SAT instances are challenging benchmarks for SAT-solving algorithms. Physicists have put forward an algorithm called Belief Propagation guided decimation that harnesses the Belief Propagation technique from AI in order to construct satisfying assignments. Experiments indicate that this algorithm far outperforms other SAT solvers such as Walksat or zChaff for $k = 3, 4, 5$. I am going to present the first rigorous analysis of BP guided decimation, showing that for larger k (the basic version of) this algorithm does not outperform other, simpler ones.

Keywords: Belief Propagation, random k-SAT, statistical mechanics

4.5 The stubborn problem is stubborn no more

Marek Cygan (University of Warsaw, PL)

We present a polynomial time algorithm for the 3-Compatible Colouring problem, where we are given a complete graph with each edge assigned one of 3 possible colours and we want to assign one of those 3 colours to each vertex in such a way that no edge has the same colour as both of its endpoints. Consequently we complete the proof of a dichotomy for the k-Compatible Colouring problem.

The tractability of the 3-Compatible colouring problem has been open for several years and the best known algorithm prior to this paper is due to Feder et al. [SODA'05], which is a quasipolynomial algorithm with a $n^{O(\log n/\log\log n)}$ time complexity.

Furthermore our result implies a polynomial algorithm for the Stubborn problem which enables us to finish the classification of all List Matrix Partition variants for matrices of size at most four over subsets of $\{0, 1\}$ started by Cameron et al. [SODA'04].

Keywords: Compatible colouring, list matrix partition, stubborn problem

Joint work of: Cygan, Marek; Pilipczuk, Marcin; Pilipczuk, Michal; Wojtaszczyk, Onufry

4.6 Listing all maximal cliques in sparse graphs in near-optimal time

David Eppstein (Univ. California - Irvine, US)

The degeneracy of an n-vertex graph G is the smallest number d such that every subgraph of G contains a vertex of degree at most d. We show that there exists a nearly-optimal fixed-parameter tractable algorithm for enumerating all maximal cliques, parametrized by degeneracy. To achieve this result, we modify the classic Bron–Kerbosch algorithm and show that it runs in time $O(dn3^{d/3})$. We also provide matching upper and lower bounds showing that the largest possible number of maximal cliques in an n-vertex graph with degeneracy d (when d is a multiple of 3 and $n \geq d + 3$) is $(n - d)3^{d/3}$. Therefore, our algorithm matches the $\Theta(d(n - d)3^{d/3})$ worst-case output size of the problem whenever $n - d = \Omega(n)$.

Keywords: Clique, backtracking, degeneracy, worst-case optimality

Joint work of: Eppstein, David; Löffler, Maarten; Strash, Darren

Full Paper: http://arxiv.org/abs/1006.5440

See also: To appear at ISAAC 2010

4.7 Parameterized Measure and Conquer: a simple example

Henning Fernau (Universität Trier, DE)

There have been quite some discussions how and under which circumstances M&C techniques can be employed.

Most published examples are of a quite intricate nature.

We present a relatively simple example for parameterized M&C in this talk, considering the problem of finding a vertex cover of size at most k in a subcubic graph.

There has been quite some work on this particular problem already.

Our proposed algorithm does not quite match the fastest ones, but it might serve as a kind of textbook example for using M&C within the analysis of parameterized algorithms: both the algorithm and its analysis are quite simple compared to published approaches.

Keywords: Parameterized algorithms; vertex cover; measure and conquer

Joint work of: Binkele-Raible, Daniel; Fernau, Henning;

4.8 A New Upper Bound for 3-SAT

Kazuo Iwama (Kyoto University, JP)

This talk gives a new randomized algorithm which solves 3-SAT in time $O(1.32113^n)$. The previous best bound is $O(1.32216^n)$ due to Rolf (J. SAT, 2006). The new algorithm uses the same approach as Iwama and Tamaki (SODA 2004), but exploits the non-uniform initial assignment due to Hofmeister et al. (STACS 2002) against Schöning's local search (FOCS 1999).

Joint work of: Iwama, Kazuo; Seto, Kazuhisa; Takai, Tadashi; Tamaki, Suguru.

4.9 A space-time tradeoff for permutation problems

Mikko Koivisto (University of Helsinki, FI)

Many combinatorial problems—such as the travelling salesman, feedback arc set, and tree-width problem—can be formulated as finding a feasible permutation on n elements. Based on the idea of "covering" the linear orders by a small family of "thin" partial orders, we show that permutation problems can be solved in time $O^*(T^n)$ and space $O^*(S^n)$ with $ST < 4$ at any $\sqrt{2} < S < 2$ and with $ST < 3.93$ at a certain S. This work was first presented at SODA'10.

Keywords: Partial order; space-time product; space-time tradeoff; travelling salesman

Joint work of: Koivisto, Mikko; Parviainen, Pekka

4.10 The power of group algebras in constrained monomial detection

Yiannis Koutis (Carnegie Mellon University - Pittsburgh, US)

Assume we are given a graph G whose vertices are colored with t different colors $C_1, \ldots, C_t$, and numbers $c_1, \ldots, c_t$. What is the complexity of detecting in G a k-path that uses at most c_i nodes from color class C_i? An $O^*(4^k)$ algorithm appeared in MFCS10. We show

that a simple modification of our group algebra-based approach for the usual k-path problem yields an an $O^*(6^{k/2})$ algorithm. We also explore applications to the weighted version of the multilinear detection problem.

Keywords: Multilinear monomial detection, max motif, weighted k-path

4.11 Circuit Complexity and Gate Elimination

Alexander S. Kulikov (Steklov Inst. - St. Petersburg, RU)

In the talk, we will first briefly describe the gate elimination method (essentially, the only known method for proving non-trivial lower bounds for unrestricted circuits) and then present two proofs of $7n/3$ and $3n$ lower bounds that are much simpler than the known proofs.

Keywords: Circuit complexity, gate elimination, lower bounds

4.12 The deficiency of clauses-sets: An interesting complexity parameter

Oliver Kullmann (University of Wales - Swansea, GB)

We start with an overview on the deficiency $\delta(F) = c(F) - n(F)$ of clause-sets (CNFs), the difference between the number of clauses and the number of variables.

For minimally unsatisfiable clause-sets F the basic fact is $\delta(F) \geq 1$ ("Tarsi's Lemma").

The classification of the layers of MU (the class of minimally unsatisfiable clause-sets) for $\delta = 1, 2, \ldots$ is an interesting research project. A basic result here is that SAT-decision is fixed-parameter tractable in the maximal deficiency $\delta^*(F)$ (the maximum of $\delta(F')$ for all subsets F' of F; if F is in MU, then we have $\delta^*(F) = \delta(F)$).

For hypergraph colouring these considerations can be transferred, where then the "base layer" is directly related to the solution of the Polya Problem.

The theoretical foundation for these considerations is given by autarky theory.

After this general information, we present a recent result, obtained in collaboration with Xishun Zhao (Guangzhou). For every lean clause-set F (not having a non-trivial autarky; this includes MU) there exists a variable occurring at most $\delta(F)+1+\log_2(\delta(F))$ many times. We conjecture that a stronger version of this bound is sharp, even for the smaller class MU.

If there is no such variable in an arbitrary F, then we can conclude that F must have a non-trivial autarky. It is an open problem whether we can find such an autarky in polynomial time.

Keywords: Deficiency, SAT, CNF, autarky, minimal unsatisfiability, lean clause-sets, variable degree

4.13 Improved algorithms for Max 2-CSP

Konstantin Kutzkov (IT University of Copenhagen, DK)

The breakthrough split and list algorithm by Williams (ICALP, 2004) solves the Max 2-CSP problem in $O(1.732^n)$ steps. However, it needs $O(1.588^n)$ memory.

Several researchers have designed algorithms solving Max 2-CSP, or special cases of it, using only polynomial space and running in time $O(2^{c_d n})$ where c_d is a constant depending on the average degree d of the input formula. The current record holders are Scott and Sorkin (Discrete Optimization, 2007) with an algorithm running in $O(2^{(1-2/(d+1))n})$ steps.

In this talk I will present two simple algorithms considerably improving the above bound. The first algorithm makes use of the best known algorithm for Max 2-CSP by Gaspers and Sorkin (SODA 2009), and the second one relies on a recent graph-theoretic result by Feige and Kogan (Journal of Graph Theory, 2010).

Keywords: Max 2-CSP; exact algorithms; polynomial space

Joint work of: Kutzkov, Konstantin; Kulikov, Alexander

4.14 Exact exponential time algorithms for Capacitated Dominating Set

Mathieu Liedloff (Université d'Orleans, FR)

Given a graph $G = (V, E)$ and a capacity function $c : V \to \mathbb{N}$, the Capacitated Dominating Set problem asks to compute a dominating set $D \subseteq V$ of minimum cardinality where each vertex $v \in D$ can dominate at most $c(v)$ neighbors.

By dynamic programming over subsets and exploiting structural properties of instances that cannot be solved fast via a maximum matching approach, we show that the problem can be solved in $O(1.8463^n)$ time.

Joint work of: Liedloff, Mathieu; Todinca, Ioan; Villanger, Yngve

4.15 Known Algorithms on Graphs of Bounded Treewidth are Probably Optimal

Daniel Lokshtanov (University of California - San Diego, US)

We obtain a number of lower bounds on the running time of algorithms solving problems on graphs of bounded treewidth. We prove the results under the Strong Exponential Time Hypothesis of Impagliazzo and Paturi. In particular, assuming that SAT cannot be solved in $(2-\epsilon)^n m^{O(1)}$ time, we show that for any $\epsilon > 0$,

- Independent Set cannot be solved in $(2-\epsilon)^{\mathrm{tw}(G)}|V(G)|^{O(1)}$ time,
- Dominating Set cannot be solved in $(3-\epsilon)^{\mathrm{tw}(G)}|V(G)|^{O(1)}$ time,
- Max Cut cannot be solved in $(2-\epsilon)^{\mathrm{tw}(G)}|V(G)|^{O(1)}$ time,
- Odd Cycle Transversal cannot be solved in $(3-\epsilon)^{\mathrm{tw}(G)}|V(G)|^{O(1)}$ time,
- for any $q \geq 3$, q-Coloring cannot be solved in $(q-\epsilon)^{\mathrm{tw}(G)}|V(G)|^{O(1)}$ time, and
- Partition Into Triangles cannot be solved in $(2-\epsilon)^{\mathrm{tw}(G)}|V(G)|^{O(1)}$ time.

Our lower bounds match the running times for the best known algorithms for the problems, up to the ϵ in the base.

Keywords: Treewidth algorithms, Lower bounds

Full Paper: http://arxiv.org/abs/1007.5450

4.16 Fixed-parameter tractability of multicut parameterized by the size of the cutset

Daniel Marx (HU Berlin, DE)

Given an undirected graph G, a collection $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ of pairs of vertices, and an integer p, the Edge Multicut problem ask if there is a set S of at most p edges such that the removal of S disconnects every s_i from the corresponding t_i. Vertex Multicut is the analogous problem where S is a set of at most p vertices. Our main result is that both problems can be solved in time $2^{O(p^3)} \cdot n^{O(1)}$, i.e., fixed-parameter tractable parameterized by the size p of the cutset in the solution. By contrast, it is unlikely that an algorithm with running time of the form $f(p) \cdot n^{O(1)}$ exists for the directed version of the problem, as we show it to be W[1]-hard parameterized by the size of the cutset.

Joint work of: Marx, Daniel; Razgon, Igor

Full Paper: http://arxiv.org/abs/1010.3633

4.17 Saving Space by Algebraization

Jesper Nederlof (University of Bergen, NO)

I will first show how to solve Subset Sum with goal variable t in $\mathcal{O}^*(t)$ time and polynomial space. Secondly, I will relate this to the Dynamic Programming algorithm of Bellman and briefly discuss generalizations and further applications of the used technique.

Joint work of: Lokshtanov, Daniel; Nederlof, Jesper

4.18 Exact Algorithms and Complexity

Ramamohan Paturi (UC San Diego, US)

Over the past couple of decades, a series of exact exponential-time algorithms have been developed with improved run times for a number of problems including Maximum Independent Set, k-SAT, and k-colorability using a variety of algorithmic techniques such as backtracking, dynamic programming, and inclusion-exclusion.

The series of improvements are typically in the form of better exponents compared to exhaustive search.

These improvements prompt several questions, chief among them is whether we can expect continued improvements in the exponent.

Is there a limit beyond which one should not expect improvement? Under suitable complexity assumptions, what can we say about the likely exact complexities of various NP-complete problems? In this talk, we examine some early results for these difficult questions.

Keywords: Exact Complexity, Satisfiability

4.19 More Lower Bounds on Spanning Trees of Low Degree Connected Graphs

Mike Robson (Université Bordeaux, FR)

Lower bounds on the number of spanning trees of low degree graphs are useful for analysing the effectiveness of memorising variants of exponential algorithms. By restricting our attention to large graphs without certain features which will never arise in this application, we are able to prove bounds close to or better than the conjectured bounds for the general case.

4.20 Slightly Superexponential Parameterized Problems

Saket Saurabh (The Institute of Mathematical Sciences - Chennai, IN)

A central problem in parameterized algorithms is to obtain algorithms with running time $f(k) \cdot n^{O(1)}$ such that f is as slow growing function of the parameter k as possible.

In particular, the first natural goal is to make $f(k)$ single-exponential, that is, c^k for some constant c.

This has led to the development of parameterized algorithms for various problems where $f(k)$ appearing in their running time is of form $2^{O(k)}$. However there are still plenty of problems where the "slightly superexponential" $f(k)$ appearing in the best known running time has remained non single-exponential even after a lot of attempts to bring it down. A natural question to ask is whether the $f(k)$ appearing in the running time of the best-known algorithms is optimal for any of these problems.

In this talk, we examine parameterized problems where $f(k)$ is $k^{O(k)} = 2^{O(k \log k)}$ in the best known running time and for a number of such problems, we show that the dependence on k in the running time cannot be improved to single exponential.

Keywords: Parameterized algorithms, ETH, Algorithmic lower bounds

Joint work of: Lokshtanov, Daniel; Marx, Daniel; Saurabh, Saket

4.21 The Exponential Time Complexity of Computing the Probability that a Graph is Connected

Nina Sofia Taslaman (IT University of Copenhagen, DK)

We show that for every probability p with $0 < p < 1$, computation of all-terminal graph reliability with edge failure probability p requires time exponential in $\Omega(m/\log^2 m)$ for simple graphs of m edges under the Exponential Time Hypothesis.

Joint work of: Husfeldt, Thore; Taslaman, Nina

4.22 Inclusion/Exclusion Branching for Partial Requirements: An algorithm for k-Set Splitting

Johan van Rooij (Utrecht University, NL)

Inclusion/exclusion branching is a way to branch on requirements imposed on problems, in contrast to the classical branching on parts of the solution. The technique turned out to be useful for finding and counting (minimum) dominating sets (van Rooij et al., ESA 2009).

In this talk, we extend the technique to the setting where one is given a set of properties and seeks (or wants to count) solutions that have at least a given number of these properties.

We focus on using the new approach combined with previous work to give a polynomial space algorithm for k-Set Splitting that improves the fastest known result significantly.

We will also apply the new idea to the fastest polynomial space algorithm for counting dominating sets, and directly obtain a polynomial space algorithm for Partial Dominating Set with the same running time up to a linear factor.

Keywords: Inclusion/Exclusion, Branching Algorithm, Measure and Conquer, Set Splitting, Partial Dominating Set

Joint work of: van Rooij, Johan; Nederlof, Jesper

See also: Jesper Nederlof and Johan M. M. van Rooij. Inclusion/Exclusion Branching For Partial Dominating Set and Set Splitting, International Symposium on Parameterized and Exact Computation, IPEC 2010

4.23 New Plain-Exponential Time Classes for Graph Homomorphism

Magnus Wahlström (MPI für Informatik - Saarbrücken, DE)

A homomorphism from a graph G to a graph H (assume both are simple, undirected graphs) is a mapping $f : V(G) \to V(H)$ such that if $uv \in E(G)$ then $f(u)f(v) \in E(H)$. The problem $\mathrm{Hom}(G, H)$ of deciding whether there is a homomorphism from G to H is NP-complete, and in fact the fastest known algorithm for the general case has a running time of $O^*(n(H)^{cn(G)})$, for a constant $0 < c < 1$ (as usual, the notation $O^*(\cdot)$ signifies that polynomial factors have been ignored). We show that in the restricted case that either G or H has cliquewidth bounded by k, then the problem can be solved in time $O^*(c_k^n)$, where c_k is a constant depending on k.

The same holds for the problem of computing whether a graph has cliquewidth at most k.

Full Paper: http://dx.doi.org/10.1007/s00224-010-9261-z

See also: Wahlström, Magnus: "New Plain-Exponential Time Classes for Graph Homomorphism." Theory of Computing Systems, 2010.

4.24 Improving exhaustive search implies superpolynomial lower bounds

Ryan Williams (IBM Almaden Center - San José, US)

The P vs NP problem arose from the question of whether exhaustive search is necessary for problems with short verifiable solutions. We do not know if even a very slight algorithmic improvement over exhaustive search is universally possible for all NP problems, and to date no major consequences have been derived from the assumption that an improvement exists.

We show that for some natural NP and BPP problems, minor algorithmic improvements over the trivial deterministic simulation already entail lower bounds such as NEXP is not in P/poly and LOGSPACE is not equal to NP.

These results are especially interesting given that similar improvements have been found for many other hard problems.

Optimistically, one might hope our results suggest a new path to lower bounds; pessimistically, they show that carrying out the modest program of finding slightly better algorithms for all search problems may be difficult (if not impossible).

4.25 Efficient Listing of General Graph Patterns

Virginia Vassilevska Williams (University of California - Berkeley, US)

Many parametrized problems ask for some special subgraph of size k in a large graph of size n. Given an algorithm for determining if a substructure exists, one can typically find an example of the substructure by self-reducibility, in $O(n)$ calls to the graph. We give simple black-box methods for reducing the running time overhead from search to decision in parameterized algorithms, which leads to listing algorithms with low amortized complexity per item. Our results are as follows.

Given an $O(f(k)n^c)$-time algorithm for detecting any graph pattern of size k in a graph on n nodes, one can produce an $O(k^2f(k)n^c)$ algorithm for finding such a pattern. The overhead of listing up to L subgraphs can also be reduced significantly by simply adapting calls to a decision oracle. If $D(n,k)$ is the time to detect a graph pattern of size k, then L such subgraphs can be listed roughly in time $O(Lk^2D(n/L^{1/k},k))$. We give applications to efficient finding and listing of k-trees, k-cliques and other k-subgraphs.

Keywords: Pattern listing

Joint work of: Williams, Virginia Vassilevska; Williams, Ryan

5 Open problems

5.1 Bandwidth in $O^*(4^n)$

by *Marek Cygan.*

Given a graph G with n vertices, an ordering is a bijective function $\pi : V(G) \to \{1, 2, \dots, n\}$. Bandwidth of π is a maximal length of an edge, i.e., $\mathrm{bw}(\pi) = \max_{uv \in E(G)} |\pi(u) - \pi(v)|$. The Bandwidth problem, given a graph G and a positive integer b, asks if there exists an ordering of bandwidth at most b.

Question Design an algorithm for the Bandwidth problem running in $O^*(4^n)$ time.

Remark Currently the best known polynomial space algorithm has $O(9.37^n)$ time complexity, whereas the best known exponential space algorithm runs in time $O(4.39^n)$ [1].

References

[1] M. Cygan, and M. Pilipczuk, *Bandwidth and Distortion Revisited*, CoRR, abs/1004.5012, 2010, http://arxiv.org/abs/1004.5012.

5.2 Parametrized counting of maximal cliques

by *David Eppstein.*

This problem concerns the relationship between the worst-case complexity of finding cliques in arbitrary graphs and in sparse graphs. We let n and m represent the number of edges in a given graph G, and d represent the *degeneracy* of G; that is, d is the smallest number for which every subgraph of G contains at least one vertex of degree at most d [3, 7, 11]. It is possible to order the vertices of G, by a linear time greedy algorithm, in such a way that each vertex has at most d later neighbors in the ordering; such an ordering is called a *degeneracy ordering.*

What we know already:

- Many algorithms are known with running times of the form $O(c^n)$ for finding the maximum clique in a graph, with varying values of c depending on whether polynomial or exponential space is allowed [4, 9, 10, 12]. Any such algorithm may be trivially converted into an algorithm for the same problem with the running time $O(nc^d)$, by calling the algorithm separately on each of the subgraphs induced by a vertex and its later neighbors in a degeneracy ordering.

- The *clique polynomial* [6], a polynomial in which the coefficient of x^i is the number of i-vertex cliques in a graph, may be trivially computed in time 2^n, and apparently there is an unpublished manuscript showing that nothing significantly faster is possible unless the strong exponential time hypothesis is false. Regardless of whether the best time bound of the form c^n has $c = 2$ or c smaller, there is again an automatic conversion to an algorithm with running time $O(nc^d)$: if one adds the clique polynomials of the subgraphs induced by a vertex and its later neighbors, the resulting sum overcounts the cliques in a predictable way (cliques of size i also contribute to the coefficients of x^j for each $j < i$) that is easily inverted.

- All maximal cliques may be listed in time $O(3^{n/3})$, optimum in the worst case since there exist graphs with this many cliques [1, 8, 13]. All maximal cliques may also be listed in time $O(dn3^{d/3})$, almost optimum since there exist graphs with $(n - d)3^{d/3}$ cliques [2]. However, although the $O(3^{n/3})$ bound and the $O(dn3^{d/3})$ bound can be shown using closely related algorithms, the conversion from one to the other is not automatic.

- All maximal cliques may be counted in time $O(c^n)$ for $c \approx 1.3642$, significantly less than the time to list them all [5]. However, there is no known algorithm parameterized by degeneracy for counting maximal cliques faster than listing them.

Question What is the best possible time bound of the form $O(nf(d))$ for computing the number of maximal cliques in a graph? Is it possible to solve this problem in time $O(nc^d)$ where c is the base of the exponent in the best time bound of the form $O(c^n)$ for counting maximal cliques, or at least to solve it more quickly than the $O(dn3^{d/3})$ bound for listing all maximal cliques?

References

[1] D. Eppstein. Small maximal independent sets and faster exact graph coloring. *J. Graph Algorithms & Applications* 7(2):131–140, 2003.

[2] D. Eppstein, M. Löffler, and D. Strash. Listing all maximal cliques in sparse graphs in near-optimal time. *Proc. 21st International Symposium on Algorithms and Computation*, 2010.

[3] P. Erdős and A. Hajnal. On chromatic number of graphs and set-systems. *Acta Mathematica Hungarica* 17(1–2):61–99, 1966.

[4] F. V. Fomin, F. Grandoni, and D. Kratsch. Measure and conquer: a simple $O(2^{0.288n})$ independent set algorithm. *Proc. 17th ACM-SIAM Symposium on Discrete Algorithm*, pp. 18–25, 2006.

[5] S. Gaspers, D. Kratsch, and M. Liedloff. On independent sets and bicliques in graphs. *Proc. 34th Int. Worksh. Graph-Theoretic Concepts in Computer Science*, pp. 171–182. Springer-Verlag, Lect. Notes. Comp. Sci. 5344, 2008.

[6] C. Hoede and X. Li. Clique polynomials and independent set polynomials of graphs. *Discrete Math.* 125(1–3):219–228, 1994.

[7] D. R. Lick and A. T. White. k-degenerate graphs. *Canad. J. Math.* 22:1082–1096, 1970.

[8] J. W. Moon and L. Moser. On cliques in graphs. *Israel J. Math.* 3(1):23–28, 1965.

[9] J. M. Robson. Algorithms for maximum independent sets. *J. Algorithms* 7(3):425–440, 1986.

[10] J. M. Robson. Finding a maximum independent set in time $O(2^{n/4})$. Tech. Rep. 1251-01, LaBRI, Université Bordeaux I, 2001.

[11] G. Szekeres and H. S. Wilf. An inequality for the chromatic number of a graph. *Journal of Combinatorial Theory*, 1968.

[12] R. E. Tarjan and A. Trojanowski. Finding a maximum independent set. *SIAM J. Comput.* 6(3):537–546, 1977.

[13] E. Tomita, A. Tanaka, and H. Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theor. Comput. Sci.* 363(1):28–42, 2006.

5.3 Edge colouring under ETH

by *Thore Husfeldt.*

The *k-edge colouring* problem asks for a mapping $f: E \to \{1, \ldots, k\}$ that "colours" the edges of a given undirected graph $G = (V, E)$ so that no pair of adjacent edges has the same colour. It is known that the smallest k for which such a colouring exists is closely related to the maximum degree d of the graph.

Algorithms with running times around $\exp(O(nd))$ exist; for example, by building the line graph of G and running a vertex colouring algorithm. It is easy to show using standard reductions that under the exponential time hypothesis no algorithm with running time $\exp(o(n))$ can exist. The gap between these two results, in particular our lack of understanding of the dependency on k (or, equivalently, d) has been pointed out several times [1].

A timid first step would be to establish a result analoguous to what is known for k-satisfiability [2]:

Question For $k = 3, 4, \ldots$, let c_k denote the infimum of the values δ for which there exists a $O(\exp(\delta n))$ algorithm for k-edge colouring. Show that c_k is an increasing sequence assuming the exponential time hypothesis.

References

[1] L. Kowalik, Edge coloring, Open Problems, Moderately Exponential Time Algorithms Dagstuhl Seminar 08431

[2] R. Impagliazzo, R. Paturi, On the complexity of k-SAT, J. Comput. Syst. Sci. 62(2):367-375 (2001)

5.4 Time-Space Tradeoff for TSP

by *Mikko Koivisto.*

The fastest algorithm for TRAVELING SALESMAN PROBLEM (TSP) runs in time $O^*(2^n)$ and space $O^*(2^n)$, where n is the number of cities [1, 3]. If only polynomial space is allowed, the fastest algorithm takes time $4^n n^{O(\log n)}$ [2].

Question Design a deterministic algorithm for TSP with running time $O^*(T^n)$, using space $O^*(S^n)$, such that $S \cdot T < 3.92$.

Remark A time-space tradeoff with $3.92 < S \cdot T < 3.93$ is achieved in [4].

References

[1] R. Bellman, *Dynamic programming treatment of the travelling salesman problem*, J. Assoc. Comput. Mach., 9 (1961), pp. 61?63.

[2] A. Björklund and T. Husfeldt, *Exact algorithms for exact satisfiability and number of perfect matchings*, Algorithmica, 52 (2008), pp. 226?249.

[3] M. Held and R. Karp, *A dynamic programming approach to sequencing problems*, J. Soc. Indust. Appl. Math., 10 (1962), pp. 196?210.

[4] M. Koivisto and P. Parviainen, *A Space-Time Tradeoff for Permutation Problems*, in Proc. of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), pp. 484-492, SIAM, 2010.

5.5 The VecMatic Number

by *Yiannis Koutis.*

Let $\mathbb{Z}_2^k$ denote the space of k-dimensional 0-1 vectors over $GF(2)$. We say that a graph G is a d-line graph, if it is the line graph of a graph H whose maximum degree is d.

Let $G = (V, E)$ be a graph, and $C : V \to \mathbb{Z}_2^k$ an assignment of $\mathbb{Z}_2^k$-vectors to the nodes of G. We call C a *k-vector coloring* if for all $v \in V$

$$C(v) \notin span(\{C(w) : w \in N(v)\})$$

In other words C is a k-vector coloring if for all $v \in V$ the vector assigned to v is linearly independent (over $GF(2)$) from the vectors assigned to its neighbors.

Definition We define the VecMatic number β_G as the minimum integer k such that G has a k-dimensional vector coloring.

While the usual notion of coloring imposes pairwise constraints between a node and its neighbors, the k-vector coloring definition imposes a neighborhood-wise constraint. An immediate observation is that $\beta_G \leq \chi_G$. A second observation is that for the case of the k-clique $\beta_G = \chi_G = k$. Several questions are open about the VecMatic Number and its relationship with the Chromatic Number χ_G.

- What is the complexity of computing β_G? The decision problem is clearly in NP. However, we don't yet have a hardness proof. The NP-hardness proof for 3-coloring doesn't seem to extend to 3-vector coloring.
- How big can β_G be in terms of χ_G? If $\beta_G = 2$ then G must be bipartite and so $\chi_G = 2$. For all $\beta_G \geq 3$, there are graphs where $\beta_G < \chi_G$. Michael Saks provided an argument showing that there are graphs where β_G is exponentially smaller than χ_G. Hopefully this will be written up soon. A precise characterization of the gap is an interesting problem.
- The gap between β_G and χ_G is perhaps more interesting when G is a d-line graph, i.e. for graphs encoding the edge-colorability problem for graphs of maximum degree d. Vizing's theorem states that a d-line graph is either d-colorable, or $(d+1)$-colorable. Holyer proved that it is NP-hard to decide between the two. It must also be the case that β_G is either equal to d or $d+1$. It can't be smaller than d because the line graph contains a d-clique, and it can't be larger than $d+1$ because it is upper bounded by the chromatic number. Does there exist a d-line graph G such that $\beta_G = d+1$? If not, what is the deterministic and randomized complexity of finding a d-vector coloring? If yes, are there line graphs where $\beta_G = \chi_G - 1$?
- Is there a planar graph G such that $\beta_G < \chi_G$?
- Assuming that the problem is NP-hard, what is its inapproximability properties? It is known that χ_G is inapproximable within a factor of $n^{1-\epsilon}$ for all ϵ, modulo mild complexity assumptions. Is β_G as hard to approximate? Also, what is the complexity parameterized with respect to k?

5.6 Surjective Coloring

by *Jan Kratochvil.*

What is the computational complexity of the following problem? Given a system of triples (X, T), does there exist a surjective 3-coloring of X such that no triple $t \in T$ has all three colors?

The difficulty lies in requiring that all three colors are used. The naive approach to try all $\binom{n}{3}$ triples of vertices as witnesses of three different colors fails – once three vertices are precolored by different colors, the question becomes NP-complete (an easy gadget construction).

The question is related to the concept of coloring so called mixed hypergraphs. It is a special case of surjective CSP.

5.7 Finding an autarky when all variables occur "often"

by *Oliver Kullmann.*

- We consider the SAT problem for conjunctive normal forms F ("clause-sets").

- The set of all minimally unsatisfiable clause-sets is denoted by $\mathcal{MU}$ (unsatisfiable clause-sets F, which become satisfiable after elimination of any clause $C \in F$).
- Autarkies for F are generalised satisfying assignments — they need only to satisfy every clause of F they "touch". For example the empty partial assignment (touching no clause) or every satisfying assignment for F (touching every clause of F) is an autarky for F.
- If the partial assignment φ is an autarky for F, then $\varphi * F \subseteq F$ (where $\varphi * F$ is the result of applying φ to F, i.e., removing satisfied clauses and falsified literals), and thus $\varphi * F$ is satisfiability-equivalent to F.
- A clause-set is called **lean** if it does not have a non-trivial (that is, non-empty) autarky. The empty clause-set is lean, every other lean clause-set is unsatisfiable.
- Minimally unsatisfiable clause-sets are lean.
- A relevant parameter here is the **deficiency** $\delta(F) := c(F) - n(F)$, the difference of number of clauses and number of variables (so $c(F) := |F|$ and $n(F) := |\operatorname{var}(F)|$).
- For $F \in \mathcal{MU}$ we have $\delta(F) \geq 1$ ("Tarsi's Lemma").
- A clause-set F is **matching-lean** iff it has no non-trivial matching autarky, which is equivalent to $\forall F' \subset F : \delta(F') < \delta(F)$. Thus in fact for non-empty matching-lean clause-sets F we have $\delta(F) \geq 1$.
- For background on autarkies and minimal unsatisfiability see [1].

Question In [2] we prove the following:
Consider a matching-lean clause-set F with $n(F) > 0$. If F does not contain a variable occurring at most $\delta(F) + 1 + \log_2(\delta(F))$ many times (since we consider variables, we count both positive and negative occurrences), then F is not lean (i.e., has a non-trivial autarky).
The question is now whether this autarky can be found in polynomial time.

Remarks

1. In [2] we prove actually a sharper bound.
2. We also discuss in [2] how to extract from F in polynomial time a satisfiable clause-set F', where then the above question of finding an autarky becomes equivalent to finding a satisfying assignment for F'.
3. The core of the proof in [2] is to prove an upper bound on the minimal variable-degree for $F \in \mathcal{MU}$, and then to lift this upper bound to lean clause-sets (so, if the upper bound is not fulfilled, then the clause-set can't be lean).

References

[1] Hans Kleine Büning and Oliver Kullmann. Minimal unsatisfiability and autarkies. In Armin Biere, Marijn J.H. Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, chapter 11, pages 339–401. IOS Press, February 2009. http://www.st.ewi.tudelft.nl/sat/handbook/

[2] Oliver Kullmann and Xishun Zhao. On variables with few occurrences in conjunctive normal forms. Technical Report arXiv:1010.5756v2 [cs.DM], arXiv, October 2010. http://arxiv.org/abs/1010.5756

5.8 Min Ones 3-SAT vs. Promise Ones 3-SAT

by *Konstantin Kutzkov.*

The famous k-SAT algorithm by Dantsin et al. [1] is based on the parameterized problem of searching for a satisfying assignment setting at most r variables to 1, Min Ones k-SAT(r). The best known upper bound for this problem in the case $k = 3$ is $O^*(2.56^r)$ [2]. Recently, Moser and Scheder [3] proposed a novel algorithm, Promise Ones 3-SAT, which finds *any* satisfying assignment in $O(2^r)$ under the promise that a satisfying assignment setting at most r variables to 1 exists. Their result is a derandomization of Schöning's algorithm for k-SAT [4]. Can we solve Min Ones 3-SAT also in time $O(2^r)$?

References

[1] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, O. Raghavan, and U. Schöning. A deterministic $(2 - 2/(k+1))^n$ algorithm for k-SAT based on local search. In *Theoretical Computer Science 289*, pages 69–83, 2002.

[2] K. Kutzkov and D. Scheder. Using CSP to improve deterministic 3-SAT. *CoRR*, abs/1007.1166, 2010.

[3] R. A. Moser and D. Scheder. A full derandomization of Schoening's k-SAT algorithm. *CoRR*, abs/1008.4067, 2010.

[4] U. Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *FOCS '99: Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, page 410, Washington, DC, USA, 1999. IEEE Computer Society.

5.9 Solving Hamiltionian cycle by branching

by *Dániel Marx.*

The Hamiltonian cycle problem can be solved in time $O^*(c^n)$ using the classical Held-Karp algorithm or the very recent algorithm of Björklund. Can we achieve a similar running time using an algorithm that is based on simple branching or backtracking? We can formalize the question the following way:

> Is there a polynomial-time algorithm that, given a graph G on $n > 3$ vertices, outputs graphs $G_1, \ldots, G_t$ such that
>
> 1. $t \leq c$ for some constant c,
> 2. G_i has at most $n-1$ vertices for every $1 \leq i \leq t$, and
> 3. G has a Hamiltonian cycle if and only if there is a $1 \leq i \leq t$ such that G_i has a Hamiltonian cycle?

Clearly, such an algorithm would imply a branching algorithm for Hamiltonian cycle with running time $O^*(c^n)$. Furthermore, it would imply a polynomial-time, polynomial-space randomized algorithm with success probability c^{-n}.

5.10 Median Finding in Exponential Families

by *Yoshio Okamoto.*

We are explicitly given a finite set X and a weight function $w\colon X \to \mathbb{R}$, and implicitly given a family $\mathcal{F} \subseteq 2^X$ of subsets of X (as a membership oracle). We want to find a set in $\mathcal{F}$ that has a median weight. Namely, for each set $S \in \mathcal{F}$ we have the associated weight $w(S) = \sum_{e \in S} w(e)$. Let's order them as $S_1, S_2, \ldots, S_{|\mathcal{F}|}$ so that S_i has no larger weight than S_j if $i < j$. The median weight is defined as $w(S_{\lceil |\mathcal{F}|/2 \rceil})$.

According to the choice of $\mathcal{F}$, we can study several problems. The simplest one is $\mathcal{F} = 2^X$. Even in this case, I don't know how hard the problem is. If we are asked to find the k-th lighest set in $\mathcal{F}$ and k is a part of the input, then the problem is NP-hard (proven by an easy reduction of the subset sum problem). During the workshop, Gerhard Woeginger asked me whether the (decision version of) the median problem belongs to NP or not. I don't have an answer to his question either.

As for the algorithm, during the workshop Marek Cygan and Fabrizio Grandoni gave me an $O^*(2^{|X|/2})$-time algorithm, as one does sort-and-search for the subset sum problem. This algorithm actually finds the k-th lighest set for any given k.

Other natural choices of $\mathcal{F}$ come from graph problems. For example, X is the edge set of an undirected graph and $\mathcal{F}$ is the family of all perfect matchings, all spanning trees, or all Hamiltonian cycles. Note that for several cases (like perfect matchings, spanning trees), there are some k-best algorithms, with which one can list all sets in $\mathcal{F}$ from the lightest to the k-th lightest, running in poly$(|X|, k)$ time. However, when $k = \lceil |\mathcal{F}|/2 \rceil$, the running time is at least as bad as $\Omega(|\mathcal{F}|)$, and not better than the brute-force enumeration.

5.11 Subexponential Time Algorithm for Feedback Arc Set in Tournaments

by *Saket Saurabh.*

A *feedback arc set* of a digraph $D = (V, A)$ is a set of arcs such that its deletion from D makes it a directed acyclic graph. It is well known that one can find a minimum sized feedback arc set of a digraph on n vertices in time $2^n n^{O(1)}$. The open problem is about tournaments. A *tournament* is an orientation of a complete undirected graph.

Open Problem Does there exists a $2^{o(n)} n^{O(1)}$ time algorithm for finding a minimum sized feedback arc set on tournaments with n vertices? Either obtain an algorithm with the desired running time or show that such an algorithm is unlikely under some reasonable complexity theory assumptions.

5.12 Time Complexity of Graph Homomorphism

by *Magnus Wahlström.*

A *homomorphism* $G \to H$ for simple, undirected graphs G, H is a mapping $f : V(G) \to V(H)$ that preserves adjacency; that is, for every edge $uv \in E(G)$, we have $f(u)f(v) \in E(H)$. The canonical example is $H = K_k$, in which $G \to H$ if and only if G is k-colorable, but many other interesting cases exist; the existence of a homomorphism $G \to H$ is NP-complete for any fix graph H which is not bipartite (and if H is bipartite, the question is equivalent to 2-coloring). See [4] for more.

With regards to time complexity, it is easily observed that the Graph Homomorphism problem (i.e., deciding whether $G \to H$ for graph G, H given as input) is an intermediate problem between the Chromatic Number problem and 2-CSP. That is, Chromatic Number is a special case of Graph Homomorphism, which is in turn a special case of 2-CSP with $n(G)$ variables and domain size $n(H)$. The former problem can be solved in $O^*(2^n)$ time [1]; for the latter, Traxler has shown that no $O^*(2^{O(n)})$-time algorithm is possible unless ETH fails [5]. The best known algorithm for Graph Homomorphism has a running time of $O^*(n(H)^{n(G)})$ [7], but it is unknown whether this is necessary. Specifically, it is an open question whether $G \to H$ can be decided in time $2^{O(n(G)+n(H))}$; this was first asked by Fomin et al. [2]. This is the main open question of this section.

Partial progress was made in [6], where it was shown that $G \to H$ can be solved in $O^*(c_k^{n(G)+n(H)})$ time if either G or H has cliquewidth at most k, where c_k depends on k.

The following open questions are raised by this, and may be interesting as intermediate questions before tackling the main question above.

- Can the Circular Chromatic Number of a graph be computed in $O^*(c^n)$ (or $O(2^n)$) time? The tools of [6] are not known to be able to handle this question.
- Can the time dependency for cases of bounded cliquewidth be improved to $O^*(2^n f(k))$ or even $O^*(2^n n^{f(k)})$?
- Related: It is unknown whether cliquewidth is FPT. In [6], an algorithm with running time $O^*((ck)^n)$ is given for deciding (exactly) whether a graph has cliquewidth at most k. Can the cliquewidth of a graph be computed in $O^*(c^n)$ time regardless of k?

References

[1] A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.

[2] F. V. Fomin, P. Heggernes, and D. Kratsch. Exact algorithms for graph homomorphisms. *Theory of Comput. Syst.*, 41(2):381 – 393, 2007.

[3] F. Fomin, P. Golovach, D. Lokshtanov, and S. Saurabh. Intractability of Clique-width Parameterizations. SIAM J. Comput., to appear.

[4] P. Hell and J. Nešetřil. *Graphs and Homomorphisms.* Oxford University Press, 2004.

[5] P. Traxler. The time complexity of constraint satisfaction. In *IWPEC*, pages 190–201, 2008.

[6] M. Wahlström. New Plain-Exponential Time Classes for Graph Homomorphism. *Theory of Comput. Syst.*, to appear.

[7] R. Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

5.13 4-Clique on 3-Uniform Hypergraphs

by *Ryan Williams.*

Given a 3-uniform hypergraph on n nodes, i.e., a collection of 3-sets over an n-element set, can a 4-clique be found in $n^{4-\varepsilon}$ time for some $\varepsilon > 0$? Here, a 4-clique is a set of four elements $\{a, b, c, d\}$ such that all four possible 3-sets over $\{a, b, c, d\}$ appear in the collection. Such an algorithm could be used to solve MAX-3-SAT and other CSP problems in less than 2^n time.

5.14 Faster algorithms for TC^0 satisfiability

by *Ryan Williams.*

A TC^0_3 circuit is a four-layer DAG with only one node at the fourth layer. There are n nodes in layer one, which are called the input nodes. All edges from layer 1 point to layer 2, all edges from layer 2 point to nodes in layer 3, all nodes in layer 3 point to the single node in layer 4. Each node is interpreted to be a MAJORITY gate. The value of a MAJORITY gate is 1 iff at least half the nodes that point to the gate are set to 1. A TC^0_3 circuit is satisfiable if there is a setting of 1's and 0's to the input nodes that makes the output node have value 1. Is there an algorithm for determining whether or not a TC^0 circuit on n inputs and n^c gates is satisfiable in $O(2^{n-\log_2 n})$ steps, for every constant c? Such an algorithm could be used to prove circuit lower bounds for depth-3 TC^0, which is open.