Open Problems

Moderately Exponential Time Algorithms
Seminar 08431

Fedor V. Fomin, Petteri Kaski, **Mikko Koivisto**, **Lukasz Kowalik**, **Dieter Kratsch**, **Yoshio Okamoto**, **Johan van Rooij**, and **Ryan Williams**

Fedor Fomin *Subgraph Isomorphism.*

In Subgraph Isomorphism problem we are given two graphs G and F, and the question is to decide if G contains F as a subgraph. There are many important special cases of this problem like Hamiltonian Cycle or Bandwidth, that can be solved in time $2^{O(n)}$, where n is the number of vertices in G. However, no such algorithm with such a running time is known for Subgraph Isomorphism. Even the existence of such an algorithm for the special case when the maximum vertex degree of F is at most 3 is open.

Johan van Rooij *Pathwidth of sparse graphs.*

Many graph problems can be solved in moderately exponential time on graphs of bounded degree. One approach is to create a path decomposition of these graphs and then solve the problem by dynamic programming. For cubic n-vertex graphs Fomin et al. proved that for large enough graphs the pathwidth can be bounded by $\frac{n}{6}$ and for maximum degree four graphs by $\frac{n}{3}$. Recent results by Rossmanith show that a number of problems can be solved in the same exponential time on tree decompositions as on path decompositions.

This leads to the natural question: does there exists similar but stronger bounds on the treewidth of bounded degree graphs for which a tree decomposition can be found in polynomial time? Also, can we derive stronger bounds on the treewidth or pathwidth of bounded degree bipartite graphs?

Johan van Rooij *Capacitated domination.* There are many NP-hard graph problems that can trivially be solved in $\mathcal{O}(2^n n^{\mathcal{O}(1)})$ by enumerating all vertex subsets, checking for each subset whether it satisfies certain properties in polynomial time, and returning the smallest or largest such subset. Many such problems such as Independent Set or Dominating Set can

Dagstuhl Seminar Proceedings 08431
Moderately Exponential Time Algorithms
http://drops.dagstuhl.de/opus/volltexte/2008/1798

actually be solved much faster, while other problems such as CAPACITATED DOMINATING SET seem to be stuck to this 2^n barrier.

In the capacitated domination problem each vertex v is supplied with a number c_v; this vertex can dominate only at most c_v vertices in its neighbourhood. It is not surprising that we cannot do better than 2^n for this problem yet (this was given as an open problem at IWPEC 2008) since the polynomial time algorithm verifying that a given vertex subset is a capacitated dominating set involves a flow algorithm or bipartite matching which is more complicated than simple neighbourhood observations as is the case for an independent set or a dominating set.

Johan van Rooij *Irredundant Set.* Consider the IRREDUNDANT SET problem. An irredundant set can be described in the following way. Consider a number of kings we want to place on the vertices our graph (the irredundant set vertices). A king claims his own vertex and all its neighbours as its own, but a king only has right of existence if he can rule some undisputed vertex of his own. For example, a king has no right of existence if all its neighbouring vertices contain a king, or if has one neighbouring king (which puts his own vertex in dispute) and all other neighbouring vertices also have some neighbour with a king. For positive examples, take any independent set or any inclusion minimal dominating set.

When looking at the 2^n vertex subset problems, the IRREDUNDANT SET problem lies in between both worlds: it can be verified that vertex subset is an irredundant set by only considering its distance two neighbourhood, while we were unable to solve this problem faster than $\mathcal{O}(2^n n^{\mathcal{O}(1)})$. Therefore, we post it as an open problem to compute the upper or lower irredundance numbers of a graph faster than $\mathcal{O}(2^n n^{\mathcal{O}(1)})$: the largest irredundant set or the smallest inclusion maximal irredundant set.

We note that the irredundance numbers are not just any numbers to compute: they have been studied extensively in graph theory before. For example, consider the (by some well known) chain:

$$ir(G) \leq \gamma(G) \leq i(G) \leq \alpha(G) \leq \Gamma(G) \leq IR(G)$$

Where $\alpha(G)$ is the cardinality of a maximum independent set of G, $i(G)$ is the cardinality of a minimum inclusion maximal independent set of G, $\gamma(G)$ is the cardinality of the minimum dominating set of G, $\Gamma(G)$ is the cardinality of a maximum inclusion minimal dominating set of G, and $ir(G)$ and $IR(G)$

correspond to the lower and upper irredundance numbers or G, respectively. Finally, irredundance is the property that makes a dominating set inclusion minimal.

Petteri Kaski *Counting edge-colorings of the complete graph.* A complete graph K_{2n} always admits a coloring of its edges with colors $\{1, 2, \ldots, 2n-1\}$ so that edges sharing an endvertex have distinct colors.

Question 1. Can one count the number of distinct edge-colorings of K_{2n} in time $2^{o(n^2)}$?

Remark. An algorithm with $O^*(2^{n(n-1)/2})$ running time follows by counting the vertex-colorings of the line graph of K_{2n} with $2n-1$ colors. See

- A. Björklund, T. Husfeldt, M. Koivisto, Set partitioning via inclusion-exclusion, *SIAM J. Comput.*, to appear.

Question 2. What is the number of edge-colorings for $2n = 16$?

Remark. For $2n = 14$ the number is

$$\begin{aligned} &13! \cdot 98758655816833727741338583040 \\ &\quad = 614972203951464612786852376432607232000. \end{aligned}$$

See

- P. Kaski, P. R. J. Östergård, There are 1,132,835,421,602,062,347 non-isomorphic one-factorizations of K_{14}, *J. Combin. Designs*, to appear. doi: 10.1002/jcd.20188

Petteri Kaski *Disjoint triples of subsets.* Let U be an n-element set. Denote by $\binom{U}{k}$ the set of all k-subsets of U. Given $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3 \subseteq \binom{U}{k}$ as input, the task is to determine whether there exists a triple $(X_1, X_2, X_3) \in \mathcal{F}_1 \times \mathcal{F}_2 \times \mathcal{F}_3$ with $X_1 \cap X_2 = X_1 \cap X_3 = X_2 \cap X_3 = \emptyset$.

Question. For which values of $1/4 \leq \alpha \leq 1/3$ and $k = \alpha n$ does there exist an algorithm with running time $O((2-\epsilon_\alpha)^n)$, with $\epsilon_\alpha > 0$ independent of n?

Remarks. A positive answer for $\alpha = 1/3$ implies an $O((2-\epsilon)^n)$ algorithm for the Hamilton Cycle/Path problem. For $\alpha < 1/4$ a positive answer is obtained by combining a trimmed fast subset convolution of f_1, f_2 with the fast intersection transform of f_3, where f_1, f_2, f_3 are indicator functions of $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$. See

- A. Björklund, T. Husfeldt, P. Kaski, M. Koivisto, Fourier meets Möbius: fast subset convolution, Proceedings of the 39th Annual ACM Symposium on Theory of Computing (San Diego, CA, June 11-13, 2007), Association for Computing Machinery, New York, 2007, pp. 67-74;

- A. Björklund, T. Husfeldt, P. Kaski, M. Koivisto, Trimmed Moebius inversion and graphs of bounded degree, Proceedings of the 25th Annual Symposium on Theoretical Aspects of Computer Science (Bordeaux, February 21-23, 2008) (S. Albers and P. Weil, Eds.), IBFI Schloss Dagstuhl, Wadern, Germany, 2008, pp. 85-96;

- A. Björklund, T. Husfeldt, P. Kaski, M. Koivisto, The fast intersection transform with applications to counting paths, arXiv:0809.2489.

Dieter Kratsch *Number of minimal dominating sets.*
Let $\mathrm{ds}(n)$ be the maximum number of minimal dominating sets in a graph on n vertices. It is known that $\mathrm{ds}(n) \geq 15^{n/6} \geq 1.5704^n$. Fomin, Grandoni, Pyatkin and Stepanov showed that $\mathrm{ds}(n) \leq 1.7159^n$ by means of a moderately exponential-time algorithm enumerating all minimal covers of a set cover instance.

- Determine $\mathrm{ds}(n)$. For which value of α is $\mathrm{ds}(n) \approx \alpha^n$?

Dieter Kratsch *Partition into increasing or decreasing subsequences.*
The problem to partition a permutation into the smallest possible number of increasing or decreasing subsequences is known to be NP-hard. When combining two old results on the problem one obtains a subexponentional time algorithm (of running time $O(n^{\sqrt{2n}})$) to solve the problem.

- Can you find a faster subexponential time algorithm for the problem?

- Is the problem fixed-parameter tractable when the parameter is the number of increasing or decreasing subsequences in the partition?

Mikko Koivisto *Reducibility among Problems in 2^n.*
For some extensively studied problems—such as TSP, Graph Coloring, #Hamiltonian Cycles, Permanent—the fastest algorithms currently known require time $2^n poly(n)$. Show that if one of these problems can be solved in

time c^n for some $c < 2$, then also the other problems in "the class" can be solved in time d^n for some $d < 2$.

Daniel Paulusma *Disconnected Cut.* Let $G = (V, E)$ be a finite, undirected, connected graph without multiple edges and without loops. Let $U \subset V$. Then $G[U]$ denotes the subgraph of G induced by U. We say that U is a *disconnected cut* if both $G[U]$ and $G[V \backslash U]$ are disconnected.

What is the computational complexity of the following problem?

DISCONNECTED CUT
Instance: A graph $G = (V, E)$ (of diameter 2)
Question: Does G have a disconnected cut?

Saying that a graph $G = (V, E)$ has a disconnected cut is equivalent to saying that

- V can be partitioned into four nonempty sets V_1, V_2, V_3, V_4 such that there is no edge $uv \in E$ with $uv \in (V_1 \times V_3) \cup (V_2 \times V_4)$;
- G allows a vertex-surjective homomorphisms to the reflexive four-cycle (a cycle on four vertices with a self-loop in every vertex);
- $\overline{G} = (V, \{uv \mid uv \notin E\})$ allows a spanning subgraph that consists of two *bicliques*, i.e., two nontrivial vertex-disjoint complete bipartite graphs.

Ryan Williams *Solving k-path in $O^*(2^k)$ time deterministically.*
Can the k-path problem be solved in $O^*(2^k)$ time, deterministically? The approach will probably have to be quite different from the known randomized algorithm, since that uses polynomial identity testing as a key subroutine.

Ryan Williams *Hybrid algorithm for vertex cover.* A *hybrid algorithm* (cf. Vassilevska-Williams-Woo, SODA'06) is a collection of three algorithms A_1, A_2, A_3, with the following curious property. A_1 is a polytime algorithm that always returns "approximate" or "exact". A_2 is a polytime approximation algorithm that only works on some inputs. A_3 is an exact (exponential) algorithm that only works on some inputs.
On each instance x of a problem,

- if $A_1(x) =$ "approximate" then $A_2(x)$ approximately solves instance x.

- if $A_1(x) =$ "exact" then $A_3(x)$ exactly solves instance x.

The overall research goal in hybrid algorithms is to find those that beat the worst case inapproximability with A_2, and get subexponential time with A_3. For example, there is a hybrid algorithm for Maximum Independent Set for all $\varepsilon > 0$ with the property that if A_2 runs then it outputs an $n^{1-\varepsilon}$-approximation in polytime, and if A_3 runs then it outputs a maximum independent set in $2^{\varepsilon' n}$ time, where ε' decreases as ε decreases. Neither of these two cases are expected to be achievable on all inputs, unless some very surprising things happen. In other words, the set of graphs for which it is hard to approximate Independent Set is a subset of those graphs for which a maximum independent set can be found rather quickly!

In general, hybrid algorithms help us get a better understanding of the relationships between hardness of approximation and hardness of exact solution. The major open problem here is to obtain a hybrid algorithm for Vertex Cover: is there a hybrid algorithm for Vertex Cover which either approximately solves within a $(2 - \varepsilon)$ factor in polynomial time, or exactly solves in $2^{\varepsilon' n}$ time, for ε' which decreases as ε decreases? Or, is there some plausible evidence that no such hybrid algorithm exists? (Does ETH fail if the algorithm exists?)

Lukasz Kowalik *Edge coloring*

In the edge coloring problem, the input is an undirected graph G of n vertices and m edges and the goal is to assign colors to edges so that incident edges get distinct colors. The number of distinct colors used should be as small as possible.

Clearly, one can reduce this problem to a vertex-coloring problem, by making a new graph G' (called *line graph*) with vertices corresponding to edges of G and such that two vertices in G' are adjacent if the relevant edges in G' are incident. Vertex-coloring G' using k colors is equivalent to edge-coloring G using k colors. It follows that we can solve the edge coloring problem in $O(2^m)$-time and space by the algorithm of Björklund, Husfeldt and Koivisto [FOCS 2006].

On the other hand, there was some work on edge-coloring cubic graphs: Eppstein and Beigel [J. Algorithms 2005] gave an $O(1.415^n)$-time algorithm and later Kowalik [WG 2006] gave an $O(1.344^n)$-time algorithm. Both these algorithm use the special properties of the edge coloring problem (in other words, they use the structure of the line graph).

The first open problem is giving an algorithm for a general case that is substantially faster than a current best vertex-coloring algorithm applied to the line graph, in other words an algorithm for general graphs which uses the structure of the line graph.

The second open problem here is the question whether one can solve the (general) edge-coloring problem in $O(c^n)$ time, for some constant c. We believe that such an algorithm does not exist, and the goal is to prove it under some complexity hypothesis (like ETH).

Yoshio Okamoto *Bicriteria Minimum-Cost Spanning Tree Problem.*

Input: A connected undirected graph $G = (V, E)$, two non-negative edge costs $c_1, c_2 \colon E \to R$, and two non-negative real numbers $b_1, b_2 \in R$.

Output: YES if there exists a spanning tree T of G such that $\sum_{e \in T} c_1(e) \leq b_1$ and $\sum_{e \in T} c_2(e) \leq b_2$; NO otherwise.

Question: Devise an algorithm for the problem above running in $O^*(c^{|E|})$ with $c < 2$.

Remark: The problem itself is known to be NP-complete (via the reduction of the partition problem) [P. Camerini, G. Galbiati, and F. Maffioli. in *Theory of Algorithms*, North-Holland, Amsterdam.] There are a pseudo-polynomial-time algorithm using the idea from Barahona and Pulleyblank [Disc. Appl. Math. 1987], and a polynomial-time approximation scheme by Goemans and Ravi [SWAT 1996] (for the definition of a polynomial-time approximation scheme for bicriteria problems, see their paper). As far as I know, the problem has not been studied in the context of moderately exponential-time algorithms. We only know the trivial algorithm that enumerates all spanning trees of a given graph.

Yoshio Okamoto *Forest Counting in Graph Classes*

Input: A undirected graph $G = (V, E)$ from a fixed graph class $\mathcal{G}$.

Output: The number of forests in G. Here, a forest means an edge-subset $F \subseteq E$ that does not embrace any cycle.

Question: Is the problem #P-complete or polynomial-time solvable when $\mathcal{G}$ is the class of cographs? What if $\mathcal{G}$ is the class of unit interval graphs?

Remark: The case of cographs was studied by Giménez, Hliněný, and Noy [SIAM J. Disc. Math. 2006)], and they gave an exact algorithm running in $O^*(\exp(|V|^{1/3}))$ time. The case of unit interval graphs was studied by Gebauer and Okamoto [Intern. J. of Foundation of Comp. Sci., to appear], and they gave an exact algorithm running in $O^*(1.9706^{|E|})$ time. They also prove that the problem is #P-complete when $\mathcal{G}$ is the class of chordal graphs.