Open Problems in Parameterized and Exact Computation — IWPEC 2008

Hans L. Bodlaender
Erik D. Demaine
Michael R. Fellows
Jiong Guo
Danny Hermelin
Daniel Lokshtanov
Moritz Müller
Venkatesh Raman
Johan van Rooij
Frances A. Rosamond

Technical Report UU-CS-2008-017
July 2008

Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
www.cs.uu.nl

ISSN: 0924-3275

Department of Information and Computing Sciences
Utrecht University
P.O. Box 80.089
3508 TB Utrecht
The Netherlands

Open Problems in Parameterized and Exact Computation — IWPEC 2008

Hans L. Bodlaender* Erik D. Demaine† Michael R. Fellows‡ Jiong Guo§
Danny Hermelin¶ Daniel Lokshtanov‖ Moritz Müller** Venkatesh Raman††
Johan van Rooij* Frances A. Rosamond‡

1 Preface (by Hans Bodlaender and Frances Rosamond)

In May 2008, the 3rd International Workshop on Parameterized and Exact Computation, IWPEC 2008, was held in Victoria, B.C., Canada. At the end of this successful workshop an *open problem session* was held, where participants presented open problems from the field of parameterized and exact computation. Here, you can read the problems presented in this open problem session, and some other problems contributed by participants of IWPEC 2008. This was the second IWPEC open problem session; the first was held at the end of IWPEC 2006 [4]. The texts here were edited and sometimes written by Frances Rosamond and Hans Bodlaender. We thank those who contributed to the success of this session, and hope that solutions to this problem will be found and find their way in the literature, perhaps in another IWPEC. Also, we hope that the problem session becomes a good IWPEC-tradition.

2 k-Origami (contributed by Erik Demaine)

In the k-Origami problem, the input is a crease pattern (a graph drawn with straight edges) on a square of paper, the parameter k is the number of vertices interior to the square of paper, and the goal is to determine whether the crease pattern is flat foldable (technically, has a flat folded state). What is the parameterized complexity of k-Origami? It is known that the problem is NP-complete and solvable in linear time for $k = 1$ [1].

*Department of Information and Computing Sciences, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, the Netherlands
†MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge, MA 02139, USA
‡Parameterized Complexity Research Unit, The University of Newcastle, Callaghan NSW 2308, Australia
§Theoretische Informatik I, Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07734 Jena, Germany
¶Department of Computer Science, University of Haifa, Mount Carmel, Haifa 31905, Israel
‖Department of Informatics, University of Bergen, N-5020 Bergen, Norway
**Mathematisches Institut, Albert Ludwigs Universität Freiburg, Eckerstrasse 1, 79104 Freiburg, Germany
††The Institute of Mathematical Sciences, C.I.T Campus, Taramani, Chennai 600113, India

3 Win/Wins (contributed by Mike Fellows)

3.1 Win/Win Gaps

A *Win/Win* algorithm is a polynomial-time algorithm that constructively links two parameters. In order for a win/win algorithm to be possible, there must be an existential fact that links them. For example, it is a theorem that:

Theorem 4 (Fellows and Langston [11]) *For any graph G, either:*

1. *G has a cycle of length at least k, or*
2. *G has treewidth at most $k-2$.*

This mathematical fact has a nice algorithmic counterpart, first noted by Fellows and Langston in [11]: There is a polynomial time algorithm that produces, for input (G, k), either:

1. a cycle in G of length at least k, or
2. a tree-decomposition of G of width at most $k-2$.

As witnessed by the complete graphs, the theorem is *best possible.* The things that make a win/win algorithm attractive include that (for example):

- The algorithm runs in polynomial time (nothing exponential in either parameter).
- One or the other outcomes might be otherwise quite expensive to produce. For example, the best known algorithm for producing a width k tree-decomposition (if one exists) runs in time $O^*(2^{35k^3})$.

Theorem 5 (Folklore.) *For any graph G, either:*

- *the minimum size of a vertex cover of G is greater than k, or*
- *G has a path-decomposition of width at most k.*

The best known win/win algorithm for Theorem B is one that either correctly determines that the vertex cover number of G greater than k, or produces a path-decomposition of width at most $2k$. Can this be improved? For example, is there a polynomial time algorithm that produces either:

- a proof that the vertex cover number of G is greater than k, or
- a path-decomposition of width at most $3k/2$ (... or $k \log k$, etc.)

or is this impossible unless something unexpected happens?

A similar situation holds for the *max leaf number $ml(G)$ versus pathwidth.*

Theorem 6 (Bienstock, Robertson, Seymour and Thomas, [2]) *For any graph G, either:*

1. *the pathwidth of G is at most k, or*

2. G has a spanning tree with at least $k+1$ leaves.

The proof of [2] does not directly yield a polynomial-time algorithm.

The best known win/win algorithm for Theorem 6 is one that in P-time produces either a spanning tree with at least $k+1$ leaves, or a path-decomposition of width at most $2k-1$. Can this be improved?

Win/win algorithms are useful in building FPT algorithms. This area might even support some interesting basic theory, as it seems that many more "kinds" of these parameter-linkage kinds of questions might be asked. If the existential theorem connects two W-hard parameters, for example, one could ask for an "FPT win/win".

6.1 Cheating the Kernelization Lower Bounds

This question is due to Jiong Guo.

Recently it has been shown that some natural parameterized problems in FPT do not admit P-time many:1 kernelization to $Poly(k)$ unless unexpected things happen (e.g., collapse of the Polynomial Hierarchy to level 3) [5]. An example of such a problem is LONG PATH, the problem of determining whether a graph G contains a path of length at least k.

But maybe we *cheat* this bad news in a P-time Turing manner. Consider LONG PATH. If G has n vertices, I propose (for example) to consider n different *localizations* of the problem to an instance of ANCHORED LONG PATH, where the input is a graph G, a vertex s of G, and the question is whether there is a path of length at least k *that begins at* s. The intuition is that maybe the "no $poly(k)$ kernel" judgment might be a bit "fragile", and if we specify a starting point for the path, then maybe we *can* get a polynomial-in-k kernelization.

"How fragile is the *bad news*?" is a very important question to ask, in general. Perhaps there is some relationship here to the *smoothed analysis* of computational complexity.

We can ask about LONG PATH: Is there a polynomial-time Turing kernelization to a "localized" version of the problem (such as the above, as one possibility), that does admit a $Poly(k)$ kernelization? Anytime that the current lower bound technology for P-time many:1 kernelization applies to a specific problem, we can meaningfully ask if we can escape that negative result by some P-time Turing "localization" for that specific problem, and this is interesting.

6.2 Empirical Parameters.

We still do not understand what is going on (epistemologically) with Karsten Weihe's well-known *Train Problem.* The main theme of parameterized complexity and algorithmics is that natural problem input distributions usually have relevant secondary structure or aspects.

But can it be that somehow, sometimes the relevant parameters are *empirical*, and how should this situation be handled?

I think this question calls for some fresh imagination in theory-building in the interstitial zone between *algorithms and complexity research* as traditionally envisioned, and the new impulse towards *algorithms engineering.*

7 Distillation of co-NP-Complete problems (contributed by Danny Hermelin)

A *distillation algorithm* for a classical problem $L \subseteq \Sigma^*$ is an algorithm that receives as input a sequence $(x_1, \ldots, x_t)$, with $x_i \in \Sigma^*$ for each $1 \leq i \leq t$, uses time polynomial in $\sum_{i=1}^{t} |x_i|$, and outputs a string $y \in \Sigma^*$ with

1. $y \in L \iff x_i \in L$ for some $1 \leq i \leq t$.
2. $|y|$ is polynomial in $\max_{1 \leq i \leq t} |x_i|$.

It is known that if any NP-complete problem has a distillation algorithm, then the polynomial hierarchy collapses [5]. The question is whether or not co-NP-complete problems have distillation algorithms. If the answer to this question is "no" under some reasonable complexity-theoretic assumption, then this will give polynomial lower-bounds of kernel sizes for many important FPT problems such as Treewidth and Cutwidth. See [5] for more details.

8 Clique-width (contributed by Daniel Lokshtanov)

This is a big project linked to many interesting open problems. Thus any non-trivial advance would be interesting. A clique-expression is a sequence of instructions that generate a graph (see definition of clique-width). (See e.g., [6, 8, 7, 10, 12, 13, 14, 15, 16, 18, 19, 22, 24, 26].) Thus every clique-expression is a string that corresponds to a graph. This means that a language of clique-expressions represents a graph family. For a regular expression ϕ that generates a language L of well-parenthesized clique-expressions, ϕ corresponds to a graph class $\Pi(\phi)$.

Example The regular expression

> Create a vertex with label 1, (Create a vertex with label 2, Join(1,2), Relabel 2 to 1)*.

represents the family of all cliques. The two main questions are the following:

(1)
Input: Graph G and regular expression ϕ generating well-parenthesized clique-expressions.
Parameter: $|\phi|$
Question: Is $G \in \Pi(\phi)$?

(2)
Input: Graph G and regular expression ϕ generating well-parenthesized clique-expressions.
Parameter: $|\phi|$
Question: Does G have an induced subgraph $H \in \Pi(\phi)$?

One can easily show that (1) is NP-complete (reduction from LINEAR CLIQUE-WIDTH) and that (2) is W-hard (reduction from CLIQUE). (See also [12].) However it is highly non-trivial to put any of these problems in XP. If (1) turns out FPT (in XP) then so does LINEAR CLIQUE-WIDTH, if (2) turns out in XP this would give a unified algorithm to recognize many hereditary graph classes (among them perfect graphs, odd-hole free graphs, chordal graphs, etc.) For (2) this is still true if we restrict the regular expression not to use the "join" operation and not to use "or".

9 Polynomial Identity Testing (contributed by Moritz Müller)

Consider the following parameterized version of POLYNOMIAL IDENTITY TESTING (PIT):

Input: an arithmetical term C.
Parameter: number of variables in C.
Question: does C compute the zero polynomial?

Is this problem fixed-parameter tractable?

An arithmetical circuit is a circuit with gates computing constants 1,0 or (binary) multiplication or addition. It is a term if and only if all gates have fan-out one. An arithmetical circuit computes in the obvious way a polynomial over the ring of integers and it is this polynomial to what the question refers to.

The problem has a W[P]-randomized solution with one-sided error, so the question is if that result can be derandomized. I know that only some very weak form of classical derandomization would suffice. The question is related to the struggle for subexponential time algorithms for the classical PIT problem.

10 Parameterizing beyond the guarantee (contributed by Venkatesh Raman)

Parameterized complexity has had tremendous success in coming up with practical algorithms for finding solutions of small size for several problems such as VERTEX COVER. But there are several situations where the solution size is large. Here are some examples:

1. The size of a minimum vertex cover in a graph on n vertices with a perfect matching is at least $n/2$.
2. The maximum number of clauses that are satisfiable in a CNF formula with m clauses is at least $m/2$.
3. The maximum number of edges in a cut in any graph on m edges is at least $m/2$.
4. The size of the maximum independent set in a planar graph on n vertices is at least $n/4$.

For all these problems, the *standard parameterized version* of the problem, (is the optimum at least k (maximization problems) or at most k (minimization problems)) is fixed-parameter tractable. This is because the guarantee on the solution size gives a simple kernel for these problems. For example, for the VERTEX COVER problem above, if $k < n/2$, answer NO, else $n \leq 2k$. In other words, the problem is interesting only when the parameter k is large, in which case, the standard brute-force exponential algorithm itself serves as a fixed-parameter tractable algorithm. However, such FPT algorithms are not practical as the exponent k, for interesting cases of the problem, is large.

One approach to address this could be to say that parameterized complexity is meaningful only when the parameter size is small. Our alternative approach is to parameterize the problems 'above the guarantee' as below.

A1 Is there a vertex cover of size at most $n/2 + k$ in the given graph on n vertices with a perfect matching?

A2 Is there a satisfying assignment satisfying at least $m/2 + k$ clauses in the given Boolean formula with m clauses?

A3 Is there a cut of size at least $m/2 + k$ in the given graph with m edges?

A4 Is there an independent set of size at least $n/4 + k$ in the given planar graph on n vertices?

Thus, by making k to be the value above the guaranteed value, we ensure that the parameter value may not be as large as before.

While problems A2 and A3 have been known to be in FPT [20], the first problem A1 has been recently shown to be in FPT by a reduction [23] to the Almost 2-Sat problem which has been shown to be in FPT [25]. The parameterized complexity of A4, and several other problems, is open.

The main open problem here is to characterize those 'above guarantee' problems that are in FPT and those that are W-hard. See [21] for more future directions and open problems related to this theme, including, some examples of problems where the above-guarantee question is W-hard. Also, for problems with guaranteed *upper* bounds, we could also parameterize below the upper bound and there are interesting open questions in this direction.

11 An exact algorithm for Capacitated Dominating Set (contributed by Johan van Rooij)

Consider the Capacitated Dominating Set problem.

Capacitated Dominating Set
Input: Undirected graph $G = (V, E)$, capacity function $c : V \to \mathbf{N}$, integer K.
Question: Is there a function $f : V \to V$, such that

- Each vertex is assigned to itself or a neighbor ($\forall v \in V : f(v) = v \vee \{v, f(v)\} \in E$.)
- A vertex gets at most its capacity many vertices assigned to it ($\forall v \in V : |\{w \in V \mid f(w) = v\}| \leq c(v)$.)
- At most K vertices have vertices assigned to it ($|\{v \in V \mid \exists w \in V : f(w) = v\}| \leq K$.)

Dom et al. [9] have investigated the fixed parameter complexity of the Capacitated Dominating Set problem and the related Capacitated Vertex Cover. Amongst others, these problems are $W[1]$-hard when parameterized by treewidth [9].

Here, we look at the time for an exact algorithm for Capacitated Dominating Set or Capacitated Vertex Cover. It is trivial to solve these problems in $O^*(2^n)$ time: enumerate all subsets of V; for each subset $W \subseteq V$, checking if W can be used as solution results in solving a generalized bipartite matching problem, which can be simply solved using flow techniques. As open problem, we have the following:

> Is there an algorithm for Capacitated Dominating Set that uses $O^*(c^n)$ time with $c < 2$?

The same problem for Capacitated Vertex Cover is also open.

12 Graph Isomorphism and Treewidth (contributed by Hans Bodlaender)

This problem was mentioned as an open problem in the STOC-talk by Kawarabayashi [17], and was discussed by participants of IWPEC.

The best known algorithm for Graph Isomorphism for graphs of bounded treewidth is dated 1990 [3], but its running time is $O(n^{k+4.5})$. It may be possible to obtain small improvements upon this time, e.g., by gaining on the matching step in the algorithm, but the main question here is:

> Does Graph Isomorphism, parameterized by treewidth, belong to FPT?

References

[1] M. W. Bern and B. Hayes. The complexity of flat origami. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'96*, pages 175–183, 1996.

[2] D. Bienstock, N. Robertson, P. D. Seymour, and R. Thomas. Quickly excluding a forest. *J. Comb. Theory Series B*, 52:274–283, 1991.

[3] H. L. Bodlaender. Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees. *J. Algorithms*, 11:631–643, 1990.

[4] H. L. Bodlaender, L. Cai, J. Chen, M. R. Fellows, J. A. Telle, and D. Marx. Open problems in parameterized and exact computation - IWPEC 2006. Technical Report UU-CS-2006-052, Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, 2006.

[5] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. To appear in Proceedings ICALP 2008, 2008.

[6] D. G. Corneil, M. Habib, J. M. Lanlignel, B. Reed, and U. Rotics. Polynomial time recognition of clique-width ≤ 3 graphs (extended abstract). In *Proceedings of the Latin American Theoretical INformatic, LATIN'2000*, pages 126–134, New York, 2000. Springer Verlag, Lecture Notes in Computer Science, vol. 1776.

[7] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique width. *Theor. Comp. Sc.*, 33:125–150, 2000.

[8] B. Courcelle and S. Olariu. Upper bounds to the clique-width of graphs. *Disc. Appl. Math.*, 101:77–114, 2000.

[9] M. Dom, D. Lokshtanov, S. Saurabh, and Y. Villanger. Capacitated domination and covering: A parameterized perspective. In M. Grohe and R. Niedermeier, editors, *Proceedings 3rd International Workshop on Parameterized and Exact Computation, IWPEC 2008*, pages 78–91. Springer Verlag, Lecture Notes in Computer Science, vol. 5018, 2008.

[10] W. Espelage, F. Gurski, and E. Wanke. Deciding clique-width for graphs of bounded treewidth. *J. Graph Algorithms and Applications*, 7:141–180, 2003.

[11] M. R. Fellows and M. A. Langston. On search, decision and the efficiency of polynomial-time algorithms. In *Proceedings of the 21st Annual Symposium on Theory of Computing*, pages 501–512, 1989.

[12] M. R. Fellows, F. A. Rosamond, U. Rotics, and S. Szeider. Clique-width minimization is NP-hard. In *Proceedings of the 38th Annual Symposium on Theory of Computing, STOC 2006*, pages 354–362, 2006.

[13] M. U. Gerber and D. Kobler. Algorithms for vertex-partitioning problems on graphs with fixed clique-width. *Theor. Comp. Sc.*, 299:719–734, 2003.

[14] M. C. Golumbic and U. Rotics. On the clique-width of some perfect graph classes. *Int. J. Found. Computer Science*, 11:423–443, 2000.

[15] F. Gurski and E. Wanke. Vertex disjoint paths on clique-width bounded graphs. *Theor. Comp. Sc.*, 359:188–199, 2006.

[16] F. Gurski and E. Wanke. Line graphs of bounded clique width. *Disc. Math.*, 307:2734–2754, 2007.

[17] K. ichi Kawarabayashi and B. Mohar. Graph and map isomorphism and all polyhedral embeddings in linear time. In *Proceedings of the 38th Annual Symposium on Theory of Computing, STOC 2006*, pages 471–480, 2008.

[18] D. Kobler and U. Rotics. Edge dominating set and colorings on graphs with fixed clique-width. *Disc. Appl. Math.*, 126:197–221, 2003.

[19] V. Lozin and D. Rautenbach. On the band-, tree-, and clique-width of graphs with bounded vertex degree. *SIAM J. Disc. Math.*, 18:195–206, 2004.

[20] M. Mahajan and V. Raman. Parameterizing above guaranteed values: Maxsat and maxcut. *J. Algorithms*, 31:335–354, 1999.

[21] M. Mahajan, V. Raman, and S. Sikdar. Parameterizing MAX SNP problems above or below guaranteed values. In H. L. Bodlaender and M. A. Langston, editors, *Proceedings 2nd International Workshop on Parameterized and Exact Computation, IWPEC 2006*, pages 38–49. Springer Verlag, Lecture Notes in Computer Science, vol. 4169, 2006. To appear in Journal of Computer and System Sciences.

[22] J. A. Makowsky and U. Rotics. On the clique-width of graphs with few P_4's. *Int. J. Found. Computer Science*, 10:329–348, 1999.

[23] S. Mishra, V. Raman, S. Saurabh, S. Sikdar, and C. R. Subramanian. The complexity of finding subgraphs whose matching number equals the vertex cover number. In *Proceedings 18th International Symposium on Algorithms and Computation, ISAAC 2007*, pages 268–279. Springer Verlag, Lecture Notes in Computer Science, vol. 4835, 2007.

[24] S. Oum and P. Seymour. Approximating clique-width and branch-width. *J. Comb. Theory Series B*, 96:514–528, 2006.

[25] I. Razgon and B. O'Sullivan. Almost 2-sat is fixed-parameter tractable. To appear in: Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP 2008), 2008.

[26] K. Suchan and I. Todinca. On powers of graphs of bounded nlc-width (clique-width). *Disc. Appl. Math.*, 155:1885–1893, 2007.