Reoptimization of Parameterized Problems (Abstract)

Hans-Joachim Böckenhauer[1], Elisabet Burjons[1], Martin Raszyk[1], and Peter Rossmanith[2]

[1] Department of Computer Science, ETH Zurich, Switzerland
hjb@inf.ethz.ch, elisabet.burjons@inf.ethz.ch, m.raszyk@gmail.com
[2] Department of Computer Science, RWTH Aachen, Germany
rossmani@informatik.rwth-aachen.de

The goal of our research is to combine the techniques of reoptimization and parametrization in order to have a better understanding of what makes a problem hard from a parameterized complexity point of view. For instance, given a solution for an instance of a parameterized problem, we look at local modifications and see if the problem becomes easier or if it stays in the same complexity class. For this, we analyze classical problems in parameterized complexity, whose complexity is well understood and classified.

While the connections between reoptimization and parameterization were not systematically explored up to now, some links were already discovered. The technique of iterative compression which was introduced by Reed, Smith, and Vetta [10] was very successfully used to design parameterized algorithms. Abu-Khzam et al. [1] looked at the parameterized complexity of dynamic, reoptimization-related versions of dominating set and other problems. Very recently, Alman, Mnich, and Williams [2] considered dynamic parameterized problems, which can be seen as a generalization of reoptimization problems.

We start by introducing the main concepts of parameterized complexity and reoptimization that we use. Introducing parameters relevant to the problems' structure and allowing non-polynomial computations on those, is a way to give more insight to the hardness of problems than just analyzing the complexity with respect to the input size. For a deeper introduction to parameterized complexity we refer the reader to [6,8].

We say that a problem L is *fixed-parameter tractable* if, given an instance (x, k), where x is the problem instance and k is the parameter, there exists an algorithm determining if $(x, k) \in L$ in time $p(|x|)f(k)$, where p is polynomial and f is an arbitrarily computable function. A *kernel* for a parameterized problem is an algorithm that transforms any instance (x, k) into an instance (x', k') with size polynomial in k, such that $(x, k) \in L$ if and only if $(x', k') \in L$. Finally, a *Turing kernelization* for a parameterized problem is a procedure that, given an instance (x, k) for a parameterized problem L, tells us if $(x, k) \in L$ in polynomial time by being allowed to query an oracle instantaneously solving instances of size polynomial in k.

Classical reoptimization tries to address the following question: We are given an optimal or nearly optimal solution to some instance of a hard optimization problem, then a small local change is applied to the instance, and we ask whether we can use the knowledge of the old solution to facilitate computing a reasonable solution for the locally modified instance. This notion was mentioned for the first time by Schäffter [11] and was for the first time successfully used for computing improved approximations in [3,4].

When combining reoptimization with parameterized complexity theory, given a parameterized problem instance (x, k) and a solution, i. e., a witness with which we can check in polynomial time that $(x, k) \in L$, we want to find out whether an instance with a local modification (lm), has a solution, i. e., if $(x_{\mathrm{lm}}, k') \in L$. We write the reoptimization version of problem L with local modification lm as lm-L and an instance for lm-L is a triple $((x, k), s, (x_{lm}, k'))$ and $((x, k), s, (x_{lm}, k')) \in$ lm-L if and only if $(x_{lm}, k') \in L$. Focussing on graph problems, we look at the four most natural local modifications of adding or deleting edges or vertices.

Bodlaender et al. [5] define the concept of *compositional parameterized problems*, specifically OR-compositional and AND-compositional problems, for both of which no polynomial kernel exists unless NP $\subseteq$ coNP/Poly. We observe that some of these problems do indeed have polynomial kernels in a reoptimization setting, where an optimal solution or a polynomial kernel is given for a locally modified instance.

OR-compositional graph problems are those, where given two graph instances for a problem, the disjoint union of these two graphs will have a solution if and only if one of them does. *AND-compositional* graph problems are those, where, given two graph instances for a problem, the disjoint union of these graphs will have a solution if and only if both of them do.

Moreover, we say a graph problem L is *monotone* if it is closed under removal of edges and vertices, that is, $(G - e, k) \in L$ if $(G, k) \in L$. Complementing this, a graph problem L is *co-monotone* if it is closed under addition of edges and vertices.

The key to our results is that, for instance, for monotone OR-compositional graph problems, finding kernels locally is sufficient to guarantee the existence of a kernel under reoptimization. We call a kernel *local* if it is a kernel for one connected component of the instance or if it guarantees to identify the existence of a solution containing a given vertex or edge. We focus on local kernels because, in a locally modified instance, any newly generated solution will always have to be in the modified component and contain the modified vertex or edge, otherwise the newly given solution would have been available before the modification.

Specifically, we prove that, given a parameterized NP-hard OR-compositional monotone graph problem L, for which local kernels around any vertex or edge can be computed, e^--L and v^--L have polynomial kernels.

Complementing this result, we prove that, if the problem is co-monotone instead, and still local kernels can be computed, then e^+-L and v^+-L have polynomial kernels.

For AND-compositional problems, we also prove that, given a parameterized NP-hard AND-compositional monotone graph problem L, for which local kernels around any vertex or edge can be computed,e^+-L and v^+-L have polynomial kernels, and, if the problem is co-monotone instead, e^--L and v^--L have polynomial kernels.

Some examples of problems satisfying these properties are e^+-Internal Vertex SubTree and v^+-Internal Vertex SubTree, e^+-k-Leaf Out Tree and v^+-k-Leaf Out Tree, and e^+-d-Clique and v^+-d-Clique.

We also show that, for the complementary local modifications, no polynomial kernels exist for these problems and also that there are some problems like Clique, for which no local kernels can be found for any of these local modifications and thus no polynomial kernel exists even in the reoptimization version.

Looking at non-compositional problems, we show that e^+-Connected Vertex Cover does not have a polynomial kernel unless NP $\subseteq$ coNP/Poly. This is proven through a reduction to Set Cover, where we show that, if a polynomial kernel existed for e^+-Connected Vertex Cover, then a polynomial compression would exist for Set Cover. Dom et al. prove in [7] that Set Cover parameterized by the size of the universe does not have a polynomial compression unless NP $\subseteq$ coNP/poly.

Finally, we take a look at Vertex Cover, a problem for which kernels exist. We construct a $2k$ kernel for e^+-Vertex Cover through a crown decomposition. The aim of this result is to prove, that the extra information provided by a solution to a neighboring instance, is useful to reduce the size of kernels for problems with well-known kernelization methods.

References

1. F. N. Abu-Khzam, J. Egan, M. R. Fellows, F. A. Rosamond, and P. Shaw. On the parameterized complexity of dynamic problems. *Theoretical Computer Science* 607, pp. 426–434, 2015.
2. J. Alman, M. Mnich, and V. Vassilevska Williams. Dynamic parameterized problems and algorithms. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP 2017)*, LIPiCS, Dagstuhl Publishing, pp. 41:1–41:16, 2017.
3. G. Ausiello, B. Escoffier, J. Monnot, and V. Paschos. Reoptimization of minimum and maximum traveling salesman's tours. In *Proc. of the 10th Scandinavian Workshop on Algorithm Theory (SWAT 2006)*, LNCS 4059, pp. 196–207, Springer-Verlag, 2006.
4. H.-J. Böckenhauer, L. Forlizzi, J. Hromkovič, J. Kneis, J. Kupke, G. Proietti, and P. Widmayer. Reusing optimal TSP solutions for locally modified input instances. In *Proc. of the 4th IFIP International Conference on Theoretical Computer Science (IFIP TCS 2006)*, pp. 251–270, 2006.
5. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423 - 434, 2009.
6. M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms.* Springer, 2015.
7. M. Dom, D. Lokshtanov, s. Saurabh. Incompressibility Through Colors and IDs. In *Proc. of Automata, Languages and Programming (ICALP 2009)*, LNCS 5555, pp.378–389, 2009.
8. R. G. Downey, and M. R. Fellows. *Fundamentals of Parameterized Complexity.* Springer-Verlag, 2013.
9. J. Flum, and M. Grohe. *Parameterized Complexity Theory (Texts in Theoretical Computer Science. An EATCS Series)*, Springer-Verlag, 2006.
10. B. A. Reed, K. Smith, and A. Vetta: Finding odd cycle transversals. *Operations Research Letters* 32(4), pp. 299–301, 2004.
11. M. W. Schäffter. Scheduling with forbidden sets. *Discrete Applied Mathematics*, 72(1), 155–166, 1997.