We Got You Covered

Demian Hespe[1], Sebastian Lamm[2], Christian Schulz[3], and Darren Strash[4]

1 **Karlsruhe Institute of Technology, Karlsruhe, Germany**
`hespe@kit.edu`
2 **Karlsruhe Institute of Technology, Karlsruhe, Germany**
`lamm@kit.edu`
3 **University of Vienna, Faculty of Computer Science, Vienna, Austria**
`christian.schulz@univie.ac.at`
4 **Hamilton College, New York, USA,** `dstrash@hamilton.edu`

Abstract

The vertex cover problem is one of a handful of problems for which *kernelization*—the repeated reducing of the input size via *reduction rules*–is known to be highly effective in practice. For our submission, we apply an initial aggressive kernelization strategy, using all known reduction rules for the problem. From there we use local search to produce a high-quality solution on the (hopefully smaller) kernel, which we use as a starting solution for a branch-and-bound solver. Our branch-and-bound solver also applies reduction rules via a branch-and-reduce scheme – applying rules when possible, and branching otherwise – though this may be toggled to omit reductions if they are not effective.

1998 ACM Subject Classification G.2.2 Graph Theory – Graph Algorithms, G.4 Mathematical Software – Algorithm Design and Analysis

Keywords and phrases kernelization, branch-and-reduce, local search

Digital Object Identifier 10.4230/LIPIcs.PACE.2019.

1 Solver Overview

A vertex cover of a graph G can be thought of as a set S of vertices of G such that every edge of G has at least one of member of S as an endpoint. A minimum vertex cover is a vertex cover having the smallest possible number of vertices for a given graph. The goal of the independent set problem is to compute a maximum cardinality set of vertices $\mathcal{I} \subseteq V$ such that no vertices in $\mathcal{I}$ are adjacent to one another. Such a set is called a *maximum independent set* (MIS). The *minimum vertex cover* problems is equivalent to the maximum independent set problem: a minimum vertex cover C in G is the complement of a maximum independent set $V \setminus C$ in G. Thus, an algorithm that solves one of these problems can be used to solve the other. For the sake of simplicity, we *write this report from the independent set perspective*. Since we also use a clique solver in our approach, note that an independent set is a clique in the complement graph.

The most efficient algorithms for finding maximum independent sets in both theory and practice use reduction rules to obtain a much smaller problem instance called a *kernel*. The kernel can then be solved quickly using exact or heuristic algorithms—or by repeatedly kernelizing recursively in the branch-and-reduce paradigm. Our solver is a combination of different kernelization techniques [3], local search [2], as well as branch-and-reduce [1, 4].

Our algorithm uses a portfolio of solvers, i.e., a branch-and-bound solver for vertex cover [1] as well as a branch-and-bound solver for the maximum clique problem [4]. Our

algorithm starts by kernelizing the graph using [3]. This algorithm uses all known reduction rules for the problem, and has an additional technique to accelerate kernelization: dependency checking to prune reductions that cannot be applied. More precisely, to compute a kernel, Akiba and Iwata [1] apply their reductions $r_1, \ldots, r_j$ by iterating over all reductions and trying to apply the current reduction r_i to all vertices. If r_i reduces at least one vertex, they restart with reduction r_1. When reduction r_j is executed, but does not reduce any vertex, all reductions have been applied exhaustively, and a kernel is found. Trying to apply every reduction to all vertices can be expensive in later stages of the algorithm where few reductions succeed. In [3], we define a scheme for checking dependencies between reductions, which allows us to avoid applying isolated vertex removal, vertex folding, and twin reductions when they will provably not succeed. After unsuccessfully trying to apply one of these reductions to a vertex v, one only has to consider v again for reduction after its neighborhood has changed. We therefore keep a set D of *viable* candidate vertices: vertices whose neighborhood has changed and vertices that have never been considered for reductions.

From there we use local search to produce a high-quality solution on the (hopefully smaller) kernel, which we use as a starting solution for a branch-and-bound solver. We use iterated local search algorithm from [2] to do this task. The algorithm is based on the notion of (j, k)-swaps. A (j, k)-swap removes j nodes from the current solution and inserts k nodes. The authors present a fast linear-time implementation that, given a maximal solution, can find a $(1, 2)$-swap or prove that none exists. We implemented the algorithm and use to find a high-quality solution of the kernel.

The solution is then used as input to a branch-and-bound solver for vertex cover. Our branch-and-bound solver also applies reduction rules via a branch-and-reduce scheme – applying rules when possible, and branching otherwise – though this may be toggled to omit reductions if they are not effective. Our implementation is similar to the algorithm of Akiba and Iwata [1]. If this is unsuccessful in a small time limit, we run the clique solver [4] on the complement of the kernel also using a small time limit. Sometimes kernelization can make the problem harder in the complement graph. Hence, if the previous call was unsuccessful we also run the clique solver on the complement of the input. Afterwards, if we still did not find a solution, we run our branch-and-bound vertex solver on the kernel using a larger time limit. If unsuccessful, we run the clique solver until the end of the time given to the program by the challenge. Note the way we integrated the different solvers is such that the overall algorithm will output solutions of the "easy" instances *quickly*, while still being able to solve hard instances.

2 Material

GitHub: `https://github.com/sebalamm/pace-2019/releases/tag/pace-2019`
DOI: `https://doi.org/10.5281/zenodo.2816116`

References

1 T. Akiba and Y. Iwata. Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover. *Theor. Comput. Sci.*, 609, Part 1:211–225, 2016.

2 D. V. Andrade, M. G. C. Resende, and Renato F. Werneck. Fast local search for the maximum independent set problem. In Catherine C. McGeoch, editor, *Experimental Algorithms, 7th International Workshop, WEA 2008, Provincetown, MA, USA, May 30-June 1, 2008, Proceedings*, volume 5038 of *Lecture Notes in Computer Science*, pages 220–234. Springer, 2008.

3 D. Hespe, C. Schulz, and D. Strash. Scalable kernelization for maximum independent sets. In *Proceedings of the Twentieth Workshop on Algorithm Engineering and Experiments, ALENEX 2018, New Orleans, LA, USA, January 7-8, 2018.*, pages 223–237, 2018.

4 C.-M. Li, H. Jiang, and F. Manyà. On minimization of the number of branches in branch-and-bound algorithms for the maximum clique problem. *Computers & OR*, 84:1–15, 2017.