Robustness in Reoptimization

Tobias Mömke*

Saarland University, Saarland Informatics Campus, Germany

Reoptimization is a framework that was introduced by Schäffter [4] to study dynamic algorithms for NP-hard optimization problems. In classical reoptimization problems, one is given an *optimal* solution to a problem instance and a local modification of the instance. The goal is to obtain a solution for the modified instance. Usually, the reoptimization version of an NP-hard problem is still NP-hard: if for the problem at hand we have a polynomial time algorithm that transforms an easy to solve instance to arbitrary valid instances using polynomially many local modifications, a polynomial time reoptimization algorithm would imply the original problem to be in P.

Instead of aiming for an optimal solution, we therefore want to compute an approximate solution. The additional information about the instance provided by the given solution plays a central role: we aim to use that information in order to obtain better solutions than we are able to compute from scratch.

For the application of reoptimization, the main obstacle is the strong initial requirement that given the solution is optimal. In particular, we cannot guarantee to maintain optimality, which limits the possibility to apply reoptimization iteratively. In this talk, we address the optimality requirement by presenting the notion of robust reoptimization which was introduced by Goyal and Mömke [3]. Instead of assuming that we are provided an optimal solution, we relax the assumption to the more realistic scenario where we are given an approximate solution with an upper bound on its performance guarantee. Formally, let Sol be a solution to the given optimization problem. Let us assume that $c(\mathrm{Sol})$ is a $(1+\epsilon)$ factor larger than the cost of an optimal solution. Then we say that a reoptimization algorithm is *robust*, if it is an approximation algorithm and its performance guarantee is $\alpha \cdot (1 + O(\epsilon))$, where α is its performance guarantee when $\epsilon = 0$. Intuitive, this definition ensures that for $\epsilon \to 0$, the performance guarantee converges smoothly towards α, independent of the given instance. We consider robustness of reoptimization algorithms to be a crucial feature, since in real world applications close to optimal solutions are much more frequent than optimal solutions.

*Research funded by Deutsche Forschungsgemeinschaft grant MO2889/1-1

We illustrate the effect by considering reoptimization of the Steiner tree problem [3, 1] and reoptimization of a TSP variant, the Latency problem [2]. We show that for Steiner tree reoptimization there is a clear separation between local modifications where optimality is crucial for obtaining improved approximations and those modifications where approximate solutions are acceptable starting points. For some of the local modifications that have been considered in previous research, we show that for every fixed $\epsilon > 0$, approximating the reoptimization problem with respect to a given $(1 + \epsilon)$-approximation is as hard as approximating the Steiner tree problem itself (whereas with a given optimal solution to the original problem it is known that one can obtain considerably improved results).

References

[1] Davide Bilò. New algorithms for steiner tree reoptimization. In *ICALP*, volume 107 of *LIPIcs*, pages 19:1–19:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[2] Wenkai Dai and Yongjie Yang. Reoptimization of minimum latency problem revisited: don't panic when asked to revisit the route after local modifications. *J. Comb. Optim.*, 37(2):601–619, 2019.

[3] Keshav Goyal and Tobias Mömke. Robust reoptimization of steiner trees. In *FSTTCS*, volume 45 of *LIPIcs*, pages 10–24. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[4] Markus W Schäffter. Scheduling with forbidden sets. *Discrete Applied Mathematics*, 72(1):155–166, 1997.