Benchmarking discrete optimization heuristics: from building a sound experimental environment to algorithm configuration

Ye, F.

Summary

Many hard optimization problems need to be solved in our daily life and research work, and various algorithms are proposed to solve different problems. Meanwhile, we lack a clear conclusion about the performance of these algorithms across different types of problems. Fortunately, benchmarking studies can help us obtain unbiased assessments of algorithms' performance.

This thesis introduces the IOHprofiler benchmarking software, which allows for an easy transition from the implementation of algorithms to the analysis and comparison of performance data. The software consists of two components: IOHexperimenter, an easy-to-use and customizable module for processing the actual experiments and generating the performance data, and IOHanalyzer, a post-processing module for compiling detailed statistical evaluations.

Benefiting from the functionalities of IOHprofiler, we can systematically perform our study of benchmarking evolutionary algorithms on discrete optimization problems. In practice, we investigated the impact of the population size and the mutation rate for the $(1+\lambda)$ EAs and the impact of the crossover probability for the $(\mu+\lambda)$ GA on OneMax and LeadingOnes. Moreover, we compared twelve heuristics and variants of a family of genetic algorithms on the twenty-five pseudo-Boolean problems. Inspired by the benchmarking results, the standard normalized bit mutation is proposed for the EAs, and non-asymptotic runtime analysis (i.e., bounds that hold for a fixed dimension rather than in big-Oh notation) is suggested for theoretical study of understanding the behavior of EAs.

Moreover, we apply the algorithm configuration methods Irace, MIP-EGO, and MIES to tune the parameters of a family of genetic algorithms. The experimental results provide insights into promising configurations of the genetic algorithm for different types of problems. In addition, we analyze the impact of the cost metric for the configuration tasks. Our results suggest that even when interested in expected running time performance (i.e., ERT), it can be preferable to use the anytime performance measure (i.e., AUC) for the configuration task.

Finally, we leverage our benchmarking data for dynamic algorithm selection, of which results show improvements obtained by switching from a genetic algorithm configuration to another one during the optimization process when compared to the static configurations.