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

Ye, F.

Chapter 8

Conclusions

In this thesis, we have developed the IOHprofiler benchmarking software, and we demonstrated the benefits of our software in benchmarking evolutionary computation methods and studying (dynamic) algorithm configuration.

In practice, we discussed in Chapter 1 the role of benchmarking in optimization and explained the demand for a new benchmarking environment. Thus, we presented our IOHprofiler benchmarking software and answered **research question 1** in Chapter 3. To address **research question 2,** we performed benchmarking studies on the evolutionary and genetic algorithms for a wide range of pseudo-Boolean optimization problems. Precisely, 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 in Chapter 4. Moreover, we compared twelve heuristics and variants of a family of genetic algorithms on the twenty-five problems provided by IOHprofiler in Chapter 5. The benchmarking studies inspired us to work on self-adaptation and algorithm configuration. Precisely, for self-adaptation, we proposed the standard normalized bit mutation for the $(1+\lambda)$ EAs and answered **research question 3** in Chapter 4. In addition, for algorithm configuration, we applied Irace, MIP-EGO, and MIES to tune the parameters of the $(\mu+\lambda)$ GA and discussed the impact of the cost metric for the configuration task in Chapter 6, answering **research question 4**. **Research question 5** was discussed in Chapter 7, where we explored the possibilities of leveraging benchmarking data for dynamic algorithm selection.

Despite the achievements described in this thesis, the goal of developing a guideline on which algorithms to favor for which kinds of the problem remains a challenging endeavor. Much research is yet to be done to answer if there is still improvement space for the design of the algorithm or how we can improve the-state-of-art algorithms. Among the possible avenues for future research, we consider the following topics particularly important: the development of IOHprofiler, the bi-objective algorithm configuration problem, advancing algorithm configuration techniques, parameter control, and dynamic algorithm selection.

- **Development of IOHprofiler** IOHprofiler will be an ongoing project, and more components will be added to bring more ideas together. The software has gained attention from the community, and we are glad to see expectations from different researchers for the future of IOHprofiler. For example, IOHprofiler, being integrated with other frameworks, contributes to the study of large scale automated algorithm design [3]. From the current development base, 1) we will consider more benchmark problems, even towards mixed-integer optimization problem and multi-objective optimization; 2) we will develop customized logging systems to support users observing algorithms' behaviour from different views of points; For example, a logger calculating the AUC values of algorithms have been applied for our study in Chapter 6; 3) novel evaluation criteria and visualization of algorithms' performance will also be a potential topic. For example, more and more criteria have been integrated into IOHanalyzer, such as the Deep Statistical Comparison analysis [59] and Shapley-values.

- **Bi-objective algorithm configuration** Based on the study of tuning $(\mu + \lambda)$ GA for minimizing ERT and maximizing AUC, respectively, we plan to study in what sense our observation that tuning for AUC can help find better configurations for ERT generalizes to other algorithm families and/or problems. Moreover, given that we can use both running time and anytime performance during the tuning process, **a bi-objective** (or even multi-objective, if considering different performance measures) optimization process might balance the advantages of the different cost metrics.

- **Advancing algorithm configuration techniques** Our results have also demonstrated that none of the configuration methods clearly outperforms all others, suggesting to either combine them or to develop **guidelines that can help users find the most suitable configuration technique for their concrete problem at hand.** We also observe that none of the techniques could find configurations that outperform or perform on par with the $(1 + 1)$ EA in several cases, indicating improvement potential for these configuration methods.

- **Parameter Control and Dynamic Algorithm Selection** The fact that optimal values of algorithm parameters change along the optimization process can be observed in the results in this thesis. We have studied self-adaptation of the mutation rate for the $(1 + \lambda)$ EAs. Also, the result in Chapter 4 has shown that the $(\mu + \lambda)$ GA benefits from dynamic crossover probability. It would certainly be interesting to extend our study to using dynamic values for the relevant parameters μ, λ, p_c, and p on more optimization problems. Interestingly, the meta-algorithm (Algorithm 7 in Chapter 4) demonstrates that RLS and EAs can be seen as two configurations of the meta-algorithm. This inspired us with the topic of *online algorithm configuration.*, which is studied in the context of dynamic algorithm selection in Chapter 7. We have studied how benchmarking data can be used to infer informed (one-shot) dynamic algorithm selection schemes for the solution of pseudo-Boolean optimization problems [167]. However, we do not obtain improvement for all the tested problems. Therefore, we will investigate efficient strategies to warm-start the algorithms by actively using the information accumulated thus far. We have used ERT values as the performance measure and as the indicator to select which algorithm combinations to execute. In future work, we will consider other performance measures, particularly those that measure the anytime performance of the algorithms.