MipConfigBench: A dataset for learning in the space of Mixed-Integer Programming algorithms[⋆]

Nick Doudchenko, Miles Lubin, Aditya Paliwal, Pawel Lichocki, and Ross Anderson

Google Research

Algorithm configuration (AC), algorithm selection (AS), and parallel algorithm portfolio (AP) are techniques for systematically choosing algorithm parameters to improve performance. Respectively, AC finds a single configuration with good performance over a family of instances, AS predicts good configurations on a per-instance basis by observing instance features, and parallel AP identifies combinations of multiple parameter settings that achieve better performance than any individual parameter setting. See [8] for a recent survey of these general techniques. In the context of mixed-integer programming (MIP), commercial solver developers have very recently begun to recognize and exploit the promise of choosing algorithm parameters based on instance features [1, 2].

Experimenting with these techniques is known to be both computationally expensive and technically challenging. [3] develops surrogate models for benchmarking AC techniques without the need for running the algorithms themselves, and [4] describes common pitfalls in connecting solvers to AC software. These challenges are particularly acute for MIP, where solving practical instances may take minutes to hours, and the commercial nature of most state-of-the-art MIP solvers imposes restrictions on who may run benchmarks and how many solves may run in parallel.

Inspired by analogous efforts in the ML community like NAS-Bench-101 [10], to address these challenges we introduce MipConfigBench, a dataset that contains solve data from over 170 million MIP solves using SCIP, a leading academic solver, after solving MIP instances from the AClib collection [6] across various meaningful configuration spaces. Alongside, we release open-source code to compute the MIP features used by [5].

Our hope is that researchers will find the MipConfigBench dataset useful when developing and benchmarking methodologies for AC and related techniques. MipConfigBench is also a dataset for training empirical performance models [7]. These are machine learning models that predict the runtime of algorithms given features of the input instances. Notably, MipConfigBench allows for easy experimentation with instance features; after computation of any features, training becomes a supervised learning problem.

Our talk covers (i) the details of the dataset, (ii) basic statistics that are easy to compute (e.g., the improvement of the single best solver (SBS) and virtual

[⋆] This work has been submitted to a conference. The rules of that conference do not allow the authors to disclose the name of the outlet.

best solver (VBS) over SCIP's default configuration), and (iii) baselines for AC using SMAC [9]. Our goal is to solicit feedback from the CPAIOR community on how such a dataset can be made more useful, and what changes or enhancements should be included in a future revision.

References

1. Berthold, T., Hendel, G.: Learning to scale mixed-integer programs. OptimizationOnline (2020)
2. Bonami, P., Lodi, A., Zarpellon, G.: A classifier to decide on the linearization of mixed-integer quadratic problems in CPLEX. OptimizationOnline (2020)
3. Eggensperger, K., Lindauer, M., Hoos, H.H., Hutter, F., Leyton-Brown, K.: Efficient benchmarking of algorithm configurators via model-based surrogates. Machine Learning **107**(1), 15–41 (Jan 2018)
4. Eggensperger, K., Lindauer, M., Hutter, F.: Pitfalls and best practices in algorithm configuration. J. Artif. Int. Res. **64**(1), 861–893 (Jan 2019)
5. Hutter, F., Hoos, H.H., Leyton-Brown, K.: Sequential model-based optimization for general algorithm configuration. In: Coello, C.A.C. (ed.) Learning and Intelligent Optimization. pp. 507–523. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
6. Hutter, F., López-Ibáñez, M., Fawcett, C., Lindauer, M., Hoos, H.H., Leyton-Brown, K., Stützle, T.: Aclib: A benchmark library for algorithm configuration. In: Pardalos, P.M., Resende, M.G., Vogiatzis, C., Walteros, J.L. (eds.) Learning and Intelligent Optimization. pp. 36–40. Springer International Publishing, Cham (2014)
7. Hutter, F., Xu, L., Hoos, H.H., Leyton-Brown, K.: Algorithm runtime prediction: Methods & evaluation. Artificial Intelligence **206**, 79–111 (2014)
8. Kerschke, P., Hoos, H.H., Neumann, F., Trautmann, H.: Automated algorithm selection: Survey and perspectives. Evolutionary Computation **27**(1), 3–45 (2019)
9. Lindauer, M., Eggensperger, K., Feurer, M., Falkner, S., Biedenkapp, A., Hutter, F.: SMAC v3: Algorithm configuration in python. https://github.com/automl/SMAC3 (2017)
10. Ying, C., Klein, A., Christiansen, E., Real, E., Murphy, K., Hutter, F.: NAS-Bench-101: Towards reproducible neural architecture search. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 7105–7114. PMLR, Long Beach, California, USA (09–15 Jun 2019)