EasyChair Preprint

№ 2634

Algorithmic Configuration by Learning and Optimization

Claudia D'Ambrosio, Antonio Frangioni, Gabriele Iommazzo and Leo Liberti

February 10, 2020

Algorithmic configuration by learning and optimization

Claudia D'Ambrosio[1], Antonio Frangioni[2], Gabriele Iommazzo[1,2], Leo Liberti[1]

[1] LIX CNRS, École Polytechnique, Institut Polytechnique de Paris, 91128, Palaiseau, France
{dambrosio, giommazz, liberti}@lix.polytechnique.fr
[2] Dip. di Informatica, Università di Pisa, 56127, Pisa, Italy
frangio@di.unipi.it

Mots-clés : *algorithm configuration, mathematical programming, machine learning, optimization solvers*

In this work we propose a methodology, based on machine learning and optimization, for selecting a solver configuration for a given instance.

General-purpose Mathematical Programming (MP) solvers have long lists of user-configurable parameters; tweaking them influences how the available algorithmic components work and interact. Therefore it can have a significant impact on the quality of the obtained solution and/or on the efficiency of the solution process. Good solvers have effective default parameter configurations, carefully selected to provide "good" performances in most cases. Furthermore, solvers may embed heuristics that try to automatically adapt the parameter configuration to the characteristics of the instance at hand. However, default/automatic parameter configurations may still be highly suboptimal with specific instances, which require a manual search for the best parameter values. The motivation for this work lies in the fact that, due to the large amount of available parameters, manual tuning is highly nontrivial and time-consuming. This setting is an instance of the Algorithm Configuration Problem (ACP) [1].

Our approach for addressing the ACP on MP solvers is based on a two-fold process:

(i) in the *Performance Map Learning Phase* (PMLP), we employ supervised Machine Learning (ML) techniques [2] in order to learn a performance function of the solver, which maps some features of the instance being solved, together with a given parameter configuration, into some measure of solver efficiency and effectiveness. Notably, the training set is obtained by running the solver, with several configurations, on instances of a specific Mixed Integer Programming (MIP) problem and measuring the integrality gap achieved within a fixed time limit;

(ii) the formal properties defining the ML methodology underlying the PMLP are translated into MP terms. The resulting formulation, together with constraints encoding the compatibility of the configurations' parameter values, is called the *Configuration Space Search Problem* (CSSP), a Mixed-Integer Nonlinear Program (MINLP) which, for a given instance, finds the configuration providing optimal performance w.r.t. the performance function. The actual implementation of the CSSP depends on the MP formulation selected to encode the learned performance function.

The main novelty of our approach lies in the fact that we explicitly model and optimize the CSSP using the mathematical description of the ML technique used to learn the performance function. This is in contrast to most of the existing algorithmic configuration approaches,

which instead employ heuristics (such as experimental design methods [3], local searches [4], genetic algorithms [5], evolutionary strategies [6] and other methods [7, 8]), that are typically unfit for scaling to a huge parameter space such as, say, that of a MILP solver (e.g. IBM ILOG CPLEX [9]).

References

[1] Eggensperger, K. and Lindauer, M. and Hutter, F., Pitfalls and Best Practices in Algorithm Configuration, CoRR, abs/1705.06058, 2017.

[2] Mohri, M. and Rostamizadeh, A. and Talwalkar, A., Foundations of Machine Learning, The MIT Press, 2012.

[3] Adenso-Díaz, B. and Laguna, M., Fine-tuning of algorithms using Fractional Experimental Design and Local Search, Operations Research, 54, 1, 99–114, 2006.

[4] Hutter, F. and Hoos, H. H. and Leyton-Brown, K. and Stützle, T., ParamILS: An Automatic Algorithm Configuration Framework, J. Artif. Int. Res., 36, 1, 267–306, AI Access Foundation, 2009.

[5] Ansótegui, C. and Sellmann, M. and Tierney, K., A Gender-based Genetic Algorithm for the Automatic Configuration of Algorithms, Proceedings of the 15th International Conference on Principles and Practice of Constraint Programming, CP'09, 142–157, Springer-Verlag, Berlin, Heidelberg, 2009.

[6] Brendel, M. and Schoenauer, M., Instance-based Parameter Tuning for Evolutionary AI Planning, Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation, GECCO '11, Dublin, Ireland, 591–598, ACM, 2011.

[7] Belkhir, N. and Dreo, J. and Savéant, P. and Schoenauer, M., Feature Based Algorithm Configuration: A Case Study with Differential Evolution, 9921, PPSN XIV, 156-165, 2016.

[8] López-Ibáñez, M. and Dubois-Lacoste, J. and Pérez Cáceres, L. and Birattari, M. and Stützle, T., The irace package: iterated racing for automatic algorithm configuration, Operations Research Perspectives, 3, 43–58, 2016.

[9] IBM, IBM ILOG CPLEX Optimization Studio CPLEX Parameters Reference, Version 12 Release 8, 2017