

Dynamic Meta-Learning

Matteo Gagliolo

IDSIA, Galleria 2, 6928 Manno (Lugano), Switzerland
University of Lugano, Faculty of Informatics,
Via Buffi 13, 6904 Lugano, Switzerland

Most solvable AI problems can be addressed by more than one algorithm; most AI algorithms feature a number of parameters that have to be set. Both choices can dramatically affect the quality of the obtained solution, and the time spent obtaining it. Algorithm Selection, or *Meta-Learning*, techniques [1, 2] typically address these questions by solving a large number of problems with each of the available algorithms, in order to learn a mapping from $(problem, algorithm)$ pairs to expected performance. The obtained mapping is later used to select and run, for each new problem, only the algorithm that is expected to give the best results.

This approach, though being preferable to the far more popular “trial and error”, poses a number of problems. It presumes that such a mapping can be learned at all, i.e., that the actual algorithm performance on a given problem will be predictable with enough precision before even starting the algorithm — often not the case with stochastic algorithms, whose performance can exhibit large fluctuations among different runs (see, e.g., [3]). It also assumes problem instances met during the training phase to be statistically representative of successive ones. For these reasons, there usually is no way to detect a relevant discrepancy between expected and actual performance of the chosen algorithm. Finally, it neglects computational complexity issues: ranking between algorithms is often based solely on the expected *quality* of the performance; and the time spent during the training phase is not even considered, although it can be large enough to cancel any practical advantage of algorithm selection.

The *Algorithm Portfolio* paradigm [4, 5] consists in selecting a *subset* of the available algorithms, to be run in parallel, with the same priority, until the fastest one solves the problem. This simple scheme is more robust, as it’s less likely that performance estimates will be wrong for all selected algorithms, but it also involves an additional overhead, due to the “brute force” parallel execution of all candidate solvers.

In our view, a crucial weakness of these approaches is that they don’t exploit any feedback from the actual execution of the chosen algorithms. We tried to move a step in this direction, introducing *Dynamic Algorithm Portfolios* [6, 7]. Instead of *first* choosing a portfolio *then* running it, we iteratively *allocate* a time slice, sharing it among all the available algorithms, and *update the relative priorities* of the algorithms, based on their current state, in order to favor the most promising ones. Instead of basing the priority attribution on performance quality, we fix a target performance, and minimize the time to reach it. To this aim, we search for a mapping from $(problem, algorithm, current\ algorithm\ state)$ triples to *expected time* to reach the desired performance quality. The mapping is obtained training a parametric model of algorithm runtime distribution. To further reduce computational complexity, we

focus on *lifelong-learning* techniques that drop the artificial boundary between training and usage, exploiting the mapping during training, and including training time in performance evaluation.

The obtained selection technique is generic, not depending on algorithm-specific properties. We present experiments with genetic algorithms [8] and satisfiability problem solvers [9].

The target of our work is to obtain a fully dynamic meta-learning agent that learns to use a set of algorithms by solving a set of problems, with minimal a-priori knowledge, and minimal performance overhead, allowed by a continuous cycle of runtime feedback and re-allocation of computational resources.

References

- [1] J. R. Rice, “The algorithm selection problem,” in *Advances in computers* (M. Rubinoff and M. C. Yovits, eds.), vol. 15, pp. 65–118, New York: Academic Press, 1976.
- [2] R. Vilalta and Y. Drissi, “A perspective view and survey of meta-learning,” *Artif. Intell. Rev.*, vol. 18, no. 2, pp. 77–95, 2002.
- [3] C. P. Gomes, B. Selman, N. Crato, and H. Kautz, “Heavy-tailed phenomena in satisfiability and constraint satisfaction problems,” *J. Autom. Reason.*, vol. 24, no. 1-2, pp. 67–100, 2000.
- [4] B. A. Huberman, R. M. Lukose, and T. Hogg, “An economic approach to hard computational problems,” *Science*, vol. 275, pp. 51–54, 1997.
- [5] C. P. Gomes and B. Selman, “Algorithm portfolios,” *Artificial Intelligence*, vol. 126, no. 1–2, pp. 43–62, 2001.
- [6] M. Gagliolo and J. Schmidhuber, “Dynamic algorithm portfolios,” in *Ninth International Symposium on Artificial Intelligence and Mathematics, Fort Lauderdale, Florida, January 2006*, 2006.
- [7] M. Gagliolo and J. Schmidhuber, “A neural network model for inter-problem adaptive online time allocation,” in *ICANN 2005 Proceedings, Part 2* (W. Duch *et al.*, eds.), pp. 7–12, Springer, 2005.
- [8] J. H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.
- [9] J. Gu, P. W. Purdom, J. Franco, and B. W. Wah, “Algorithms for the satisfiability (SAT) Problem: A survey,” in *Satisfiability Problem: Theory and applications* (D.-Z. Du *et al.*, eds.), pp. 19–152, American Mathematical Society, 1997.