Open Algorithm Selection Challenge 2017
Setup and Scenarios

Marius Lindauer
lindauer@cs.uni-freiburg.de
Jan N. van Rijn
vanrijn@cs.uni-freiburg.de
Department of Computer Science, University of Freiburg, Germany

Lars Kotthoff
larsko@uwyo.edu
Department of Computer Science, University of Wyoming, USA

Editors: Marius Lindauer, Jan N. van Rijn and Lars Kotthoff

Abstract

The 2017 algorithm selection challenge provided a snapshot of the state of the art in algorithm selection and garnered submissions from four teams. In this chapter, we describe the setup of the challenge and the algorithm scenarios that were used.

Keywords: Algorithm Selection, Competition

1. Introduction

In many areas of AI, tremendous advances have been achieved in the last decades. Approaches often leverage problem specific characteristics for high performance; they specialize to particular problem instances. One way of leveraging this complementarity between algorithms are algorithm portfolios combined with a selector that chooses the best algorithm for a given instance – the algorithm selection problem (Rice, 1976). Formally, given a portfolio of algorithms \( \mathcal{A} \in \mathcal{P} \), a set of instances \( \mathcal{I} \) and a performance metric \( m : \mathcal{P} \times \mathcal{I} \rightarrow \mathbb{R} \) (e.g., runtime), the algorithm selection problem is to determine a mapping \( s : \mathcal{I} \rightarrow \mathcal{P} \) from an instance \( i \in \mathcal{I} \) to an algorithm \( A \in \mathcal{P} \) such that the performance across all instances is maximized (w.l.o.g):

\[
\max_s \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} m(s(i), i)
\] (1)

A common approach to the algorithm selection problem is to characterize the instances by instance features (Nudelman et al., 2004; Brazdil et al., 2008; Xu et al., 2011; Hoos et al., 2014; Fawcett et al., 2014) and use machine learning to learn the mapping \( s \).

Many different algorithm selection systems have been proposed since. Modifications of the original algorithm selection problem include for example the use of pre-solving schedules (Xu et al., 2008), per-instance algorithm schedules (Amadini et al., 2014b) or parallel portfolio selection (Lindauer et al., 2015). In addition, different machine learning approaches to learn \( s \) were proposed, e.g. regression models (Nudelman et al., 2004; Xu et al., 2008), k-nearest neighbor (Leite and Brazdil, 2005; Kadioglu et al., 2011; van Rijn et al., 2015a), pair-wise cost-sensitive random forests (Xu et al., 2011), stacked models (Kotthoff, 2012;
Samulowitz et al., 2013) and dynamic portfolios (van Rijn et al., 2015b). For a thorough overview on algorithm selection, we refer the interested reader to Kotthoff (2014).

ASlib (Bischl et al., 2016) is a benchmark library for algorithm selection that collects data from the literature to provide a reproducible way of comparing different approaches and evaluating the state of the art. It enabled the first challenge on algorithm selection in 2015 (Kotthoff et al., 2017), and the open algorithm selection challenge (OASC) in 2017. In this document, we describe the setup of the OASC and the way we selected a new set of interesting algorithm selection benchmarks for it.

2. Setup

The series of algorithm selection challenges is built upon the algorithm selection library (ASlib; Bischl et al. (2016)). An ASlib scenario contains pre-computed results for a portfolio of algorithms on a set of instances (e.g., a SAT instance or a Machine Learning dataset); i.e. \( m(A, i) \) is known for all pairs of \( A \in \mathcal{P} \) and \( i \in \mathcal{I} \). Furthermore for each instance, a set of pre-computed instance features are available, as well as the time required to compute the feature values. Having access to this data (and additional meta-data such as cutoff times) enables algorithm selection researchers to perform reproducible comparisons of approaches. ASlib distinguishes between two types of scenarios: runtime scenarios and quality scenarios. In runtime scenarios the goal is to minimize the time to select an algorithm that solves all instances (e.g. SAT, TSP), whereas in quality scenarios the goal is to find the algorithm that obtains the highest score according to some metric (e.g. Machine Learning). The main practical difference between the two types of scenarios is that the cost of feature computation adds overhead in the former, but not in the latter case. Part of the data for a scenario was given to participants to train their approaches on and another part was held out to enable verification and a fair comparison.

The main differences between the 2017 and the 2015 algorithm selection challenges are as follows:

1. In 2015, all scenarios were known, but the splits into training and test instances were unknown. In 2017, the participants had access to performance and feature data on training instances (2/3), and only the instance features for the test instances (1/3).

2. In addition, new scenarios that had not been published as part of ASlib before were included in the 2017 evaluation. All scenarios were obfuscated by replacing scenario, algorithm, instance, and feature names and multiplying all performance and feature values by 4.

3. In 2015, the participants submitted their algorithm selection systems which were trained and run by the organizers. In 2017, participants submitted their predictions for the test set.

4. The 2017 challenge allowed arbitrary schedules of feature computation and algorithm steps.

5. In 2017, each team was allowed a maximum of two submissions.
We used the closed gap metric to compare the performance of algorithm selection systems across different scenarios, as in the 2015 challenge. Given the optimal performance of the virtual best solver (oracle) $m_{VBS}$, the baseline performance of always selecting the best average solver $m_{SBS}$ (single best solver), and the algorithm selection system at hand $m_s$, the closed gap for an algorithm selection benchmark is defined as:

$$\frac{m_{SBS} - m_s}{m_{SBS} - m_{VBS}}$$

In this metric, 1.0 corresponds to a perfect score (i.e. the algorithm selection system always selects the best algorithm for each instance and does not generate overhead due to instance feature computation) and 0.0 corresponds to the baseline (i.e. the single best solver). A value of less than 0.0 indicates that the algorithm selection system is worse than the single best solver, i.e. chooses algorithms that perform worse than it.\(^1\)

3. Algorithm Selection Scenarios

Apart from giving the snapshot of the state of the art in algorithm selection at the time, the algorithm selection challenge is also an opportunity to collect new scenarios for ASLib. We build new scenarios for recent competitions, the CSP Minizinc Competition (Stuckey et al., 2014), the MaxSAT Evaluation (Argelich et al., 2008), Mittelmann’s annual MIPLib evaluation (Koch et al., 2011), the QBF evaluation (Pulina, 2016), and SAT03–16.indu, which covers the international SAT competition from 2003 to 2016 (Balyo et al., 2017). The instance features for all of these scenarios were computed with publicly-available software (BNSL (Malone et al., 2018), CSP (Amadini et al., 2014a), Machine Learning (Pfahringer et al., 2000; Sun and Pfahringer, 2013; van Rijn, 2016), MaxSAT (Ansótegui et al., 2016), MIP (Leyton-Brown et al., 2009), QBF (Pulina and Tacchella, 2010), SAT (Xu et al., 2008; Alfonso and Manthey, 2014) and TTP (Wagner et al., 2017)). The runtimes of the algorithms and the feature computation costs were not measured on the same hardware. However, since the feature costs are typically quite small compared to the algorithm runtimes, the estimation of an algorithm selection system’s performance should be quite close to its real performance.

Table 1 shows the variety of different algorithm selection scenarios we used. We collected scenarios from 8 application different domains and with different characteristics, ranging across different numbers of algorithms (5-31) and instances (100-9720). The 2017 challenge included scenarios with solution quality as the performance metric for the first time.

To study the effect of small changes between scenarios, we included two pairs of very similar scenarios. CSP-Minizinc-Obj-2016 and CSP-Minizinc-Time-2016 consider the same algorithms, instances and features, but the performance metric is different. In MAXSAT-PMS-2016 and MAXSAT-WPMS-2016, the algorithms and instances are different, but the features are the same and the instances are typically considered to be quite similar.

The remainder of this volume gives the descriptions of the submissions.

Table 1: Overview of algorithm selection scenarios used in 2017 showing the alias in the
competition, the number of algorithms $|A|$, the number of instances $|I|$, the number
of instance features $|F|$, whether costs for feature computation are available (F-
Costs), the performance objective and for runtime scenarios, the speedup of the
virtual best solver (VBS) over the single best solver ($m_{SBS}/m_{VBS}$). Scenarios
marked with an asterisk were available in ASlib before the challenge.

Acknowledgements

We thank Rolf-David Bergdoll for collecting the data for the new algorithm selection benchmarks. M. Lindauer acknowledges funding by the DFG (German Research Foundation) under Emmy Noether grant HU 1900/2-1. J. N. van Rijn acknowledges funding by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme under grant no. 716721. Lars Kotthoff thanks Marius for including him.

References


R. Amadini, M. Gabbielli, and J. Mauro. An enhanced features extractor for a portfolio of

R. Amadini, M. Gabbielli, and J. Mauro. SUNNY: a lazy portfolio approach for constraint

C. Ansótegui, J. Gabàs, Y. Malitsky, and M. Sellmann. Maxsat by improved instance-


