EKAmBa: Realtime Configuration of Algorithms with Multi-armed Bandits

Jasmin Brandt
University of Paderborn
jasmin.brandt@uni-paderborn.de

Elias Schede
Bielefeld University
elias.schede@uni-bielefeld.de

Kevin Tierney
Bielefeld University
kevin.tierney@uni-bielefeld.de

Eyke Hüllermeier
University of Munich (LMU)
eyke@lmu.de

***Abstract*—Algorithms and solvers often contain many free parameters that influence their behavior and the quality of their outputs. Thus, finding a good parameter configuration for an algorithm can lead to faster performances or higher quality solutions, and has developed into an important field in machine learning. The search for suiting parameters however is difficult due to large or even infinite configurations spaces. Moreover, existing approaches often assume that a training set is available up front. This does not necessarily hold for certain real world applications where instances arrive in sequence one after another and change over time, rendering previously found configurations outdated. In our work, we tackle this problem by means of multi-armed bandit methods. These are used to preselect a promising subset of configurations on a per-instance basis as they arrive, taking instance features into account without requiring a training set.**

I. Introduction

Algorithm configuration (AC) is the task of searching for a parameter configuration from a configuration space of a given algorithm with the goal of finding a single configuration that optimizes the algorithm's performance on a given distribution of inputs. Usually, the performance is measured either in terms of minimizing the average runtime until the algorithm returns a solution or maximizing the quality of the returned solution. Furthermore, there are two main AC scenarios: offline AC and realtime AC. While the offline AC scenario configures by means of the complete instance training set and aims at finding a configuration that generalizes well across the whole distribution of problem instances, we focus on realtime algorithm configuration (RAC), which can be regarded as an online version of AC. In the RAC scenario, problem instances arrive sequentially, and the goal is to find the best configuration in each time step to solve the problem instance currently under consideration. This setting has added challenges compared to the offline case, since it allows for drift in the distribution of the problem instances over time and does not require a set of training instances up front. AC has a high relevance in the context of trustworthy AI, because some algorithms are only usable, and some problem instances are only solvable, in an acceptable runtime with an appropriate parameter configuration. Thus, AC leads to more robust and secure algorithms and increases trust by providing procedures that choose parameters in a structured way that make manual, error prune parameter settings obsolete.

II. Problem formulation

For defining the RAC scenario, let $\mathcal{A}$ be the algorithm for which we want to find an optimal parameterization with the parameter space Λ, let $\mathcal{P}$ be a probability distribution over the set of problem instances Π, and $\pi_t \in \Pi$ be the problem instance seen in time step $t \in \mathbb{N}_{>0}$. Furthermore, we denote the performance (e.g., the runtime) of algorithm $\mathcal{A}$ with parameter configuration $\lambda \in \Lambda$ on instance $\pi \in \Pi$ as $\mathcal{A}(\lambda, \pi)$. The goal is to find an optimal configuration λ_t^* in each time step $t \in \mathbb{N}_{>0}$:

$$\lambda_t^* \in \text{argmin}_{\lambda \in \Lambda} \mathcal{A}(\lambda, \pi_t).$$

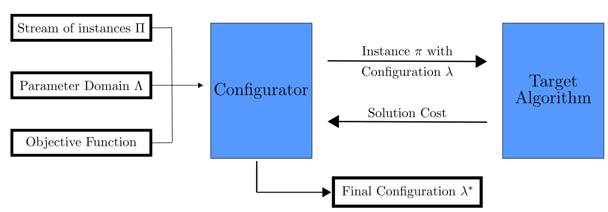

Fig. 1. Illustration of the configuration process and the interaction between configurator and target algorithm

III. RAC with bandits

In practice, we often have a large or even infinite parameter space Λ. Obviously, in this case, it is not possible to try out all possible configurations to find the one that performs best after a new problem instance π_t has arrived. Thus, we need a promising, preselected subset, from the pool of all possible configurations that fits within our computational resources. For example, we can run these configurations in parallel to find the best one, and therefore a solution to the instance in a timely manner.

a) Preselection Bandits: In the preselection bandit scenario (Bengs and Hüllermeier, 2019), we are given a set of K possible arms, which we denote with their indices $\{1, \dots, K\}$ in the following. In each time step $t \in \mathbb{N}$, the learner now has to preselect a subset $s_t \subseteq \{1, \dots, K\}$ and observes afterwards the final choice of the environment from this preselection s_t. The preselection bandits can be extended to be *contextual* preselection bandits (Mesaoudi-Paul et al., 2020a), in which the learner observes some context information before making the preselection.

b) Contextual Preselection under the Plackett-Luce Model (CPPL): Our goal is to combine principles of existing RAC approaches such as ReACT (Fitzgerald et al., 2014) and ReACTR (Fitzgerald et al., 2015) with preselection bandits. An existing method of contextual preselection bandits that has been applied to pool-based RAC is CPPL (Mesaoudi-Paul et al., 2020b). For a current problem instance π_t at time step $t \in \mathbb{N}$, the preselection bandits in CPPL choose, under consideration of the features of the instance, a preselection $s_t = \{\lambda_{t,1}, \dots, \lambda_{t,S}\}$ of a predefined size S from a pool of candidate configurations. These S configurations are then executed in parallel until the first one terminates. Besides returning the found solution for the instance, the winner information is afterwards used to update the preselection bandit model and to update the pool of candidate configurations from which the preselection bandits must choose the next subset s_{t+1} for a new instance π_{t+1}. Moreover, pool updates are performed by first discarding existing candidate configurations based on estimated bounds and second creating new configurations by applying genetic operators on the top ranked configurations.

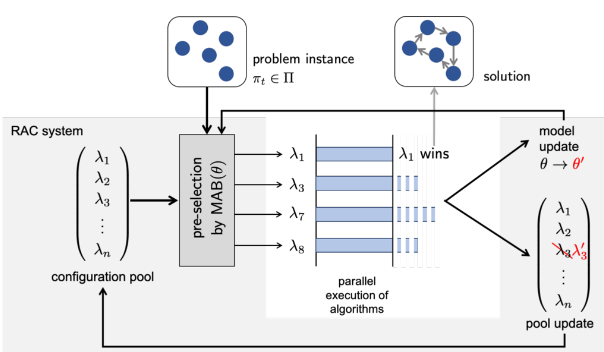

Fig. 2. Concept of the pool-based RAC with preselection bandits.

IV. Limitations and Outlook

There are some limitations of the state-of-the-art that provide possible directions to improve and extend the approach of CPPL in the context of RAC. For example, a common problem RAC methods face is the so-called burn-in phase. Since the instances arrive only step by step, the realtime algorithm configurator has not seen many instances in the first iterations and thus its outputs in the first iterations usually perform poorly. In addition, the upper confidence bound (Auer and Ortner, 2010) method by which the preselection bandits estimate the quality of the K possibilities may not be the best approach. In fact Thompson sampling (Ortega and Braun, 2014), is often able to outperform upper confidence bound methods and may also be better suited for our application. Lastly, the feature representation used to provide the contextual information to the bandits may leave room for improvement.

References

Peter Auer and Ronald Ortner. UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1):55–65, 2010. ISSN 15882829. doi: 10.1007/s10998-010-3055-6.

Viktor Bengs and Eyke Hüllermeier. Preselection Bandits under the Plackett-Luce Model. 2019. URL http://arxiv.org/abs/1907.06123.

Tadhg Fitzgerald, Yuri Malitsky, Barry O'Sullivan, and Kevin Tierney. React: Real-time algorithm configuration through tournaments. In *Seventh Annual Symposium on Combinatorial Search*, 2014.

Tadhg Fitzgerald, Yuri Malitsky, and Barry O'Sullivan. Reactr: Realtime algorithm configuration through tournament rankings. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Adil El Mesaoudi-Paul, Viktor Bengs, and Eyke Hüllermeier. Online Preselection with Context Information under the Plackett-Luce Model. pages 1–14, 2020a. ISSN 2331-8422. URL http://arxiv.org/abs/2002.04275.

Adil El Mesaoudi-Paul, Dimitri Weiß, Viktor Bengs, Eyke Hüllermeier, and Kevin Tierney. Pool-based realtime algorithm configuration: A preselection bandit approach. In *International Conference on Learning and Intelligent Optimization*, pages 216–232. Springer, 2020b.

Pedro A. Ortega and Daniel A. Braun. Generalized Thompson sampling for sequential decision-making and causal inference. *Complex Adaptive Systems Modeling*, 2(1), 2014. ISSN 21943206. doi: 10.1186/2194-3206-2-2.