Joel Greenyer, Malte Lochau, and Thomas Vogel (Hrsg.): Explainable Software for Cyber-Physical Systems (ES4CPS), GI-Dagstuhl Seminar 19023, January 2019,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2019 11

Automated Experimentation for Online Learning and Adaptation

Ilias Gerostathopoulos[1]

1 ES4CPS problem

Cyber-physical systems (CPS) are typically large systems composed of a number of independently developed and tested components that may opportunistically collaborate with each other. Consider, for example, case of several robots collaborating in pushing a door open or passing through a narrow passage. As another example, consider the complex interactions of several vehicles in a traffic network (e.g. a city or a highway) that need to collaborate in order to reduce trip times, fuel consumption and CO2 emissions. One problem in large CPS is that classical engineering approaches for developing complex systems that follow the specify-develop-test-analyze-deploy loop may not apply. A particular problem exists in the analyze phase that, traditionally, precedes the deployment phase: complex large CPS are difficult to model and analyze before deploying them in production environments, where the whole range of interactions (including emergent ones) is manifested. At the same time, complex CPS need to be explainable and predictable in their behavior. To make things worse, they may need to adapt at runtime to deal with changes in their runtime context (e.g. in the robotic case, consider an extra obstacle that makes the door harder to open).

A particular problem I am interested in is that of optimizing a traffic router used by a number of cars in a city (published as "CrowdNav" exemplar at SEAMS 2017[2]). The problem consists of selecting the values for a number of (numeric) parameters or a router used by all cars in a city in order to minimize the trip time of the cars. The complexity lies in that the cars have random destinations and have complex interactions in the physical world resulting in traffic slowdowns, jams and accidents. Moreover, the number of cars in the city may increase or decrease following reality in which traffic the varies within a single day. One question is how to find a solution to the above optimization problem in the fastest way, since the environment may change before the optimization finishes. Another challenge is how to ensure the stability of the solution. A third challenge lies in handling complaints of drivers when the optimization algorithm tries risky configurations.

[1] Technical University Munich, Software and Systems Engineering, Boltzmannstrasse 3, 85748 Garching bei München, Germany gerostat@in.tum.de

[2] S. Schmid, I. Gerostathopoulos, C. Prehofer, and T. Bures, "Self-Adaptation Based on Big Data Analytics: A Model Problem and Tool," in Proc. of SEAMS 2017, `https://github.com/iliasger/CrowdNav`

2 ES4CPS solution

With respect to CrowdNav, we have so far applied a combination of factorial analysis and Bayesian optimization in order to optimize trip durations with a very small number of tries[3]. In this solution, the traffic system together with the router is viewed as a black box with the seven parameters of the router as input and the trip duration of cars as a single output. First, a factorial design is applied in order to examine the effect of different, pre-defined, combinations of parameter values to the output. Then, only a subset of these parameters, the ones with the stronger, statistically significant, effect on the output, are selected to be optimized via Bayesian optimization. Bayesian optimization essentially performs a search in the space of input parameters in order to learn which combination minimizes the value of the output. The current solution does not take into account changes in the environment that may lead to new optimization rounds. Moreover, it does not cater for reusing knowledge about the effect of inputs to outputs across different system situations (e.g. low traffic, medium traffic, high traffic). These are two highly interesting topics that can be investigated further.

Broadening the scope, I envision a method that automates the experimentation with a system in production by learning system models from data (in the case of CrowdNav, important data are system outputs, i.e. trip durations) and using them for adapting to runtime changes. In order to be used in real-life settings, the method needs to provide guarantees w.r.t. the use of the learned models and support rump-up and abortion of experiments, similar to A/B testing, to deal with experiments that have high cost. So far we have investigated the use of statistical methods such as t-test and factorial ANOVA, to provide statistical guarantees[4].

3 Contributing Expertise

I have working knowledge in statistical testing and Bayesian optimization. I have expert knowledge of the CrowdNav testbed which I have co-developed, maintained, and extended. Finally, I have co-developed a tool that allows one to specify the inputs and the outputs of a system-to-be-optimized and run different experimentation pipelines on it[5].

4 External Expertise needed

Expertise in machine learning (and in particular transfer learning), data science, statistical methods, optimization, and search methods would be very useful in tackling some of the hard challenges of automated experimentation. As another interesting topic, expertise in requirements engineering would be useful in determining the optimization goals and criteria and the configuration points in a system-to-be-optimized.

[3] I. Gerostathopoulos, C. Prehofer, and T. Bures, "Adapting a System with Noisy Outputs with Statistical Guarantees," in Proc. of SEAMS 2018, 2018, pp. 58–68

[4] As above

[5] `https://github.com/iliasger/OEDA`