Poster: Automatic Parameter Exploration for Low-Power Wireless Protocols

Hassaan Hydher , Markus Schuß, Olga Saukh, Carlo Alberto Boano, and Kay Römer
Institute of Technical Informatics, Graz University of Technology, Austria
{m.h.mohamedhyder, markus.schuss, saukh, cboano, roemer}@tugraz.at

Abstract

We present APEX, a novel parameter exploration framework for low-power wireless protocols. APEX can autonomously derive an optimized set of parameters allowing a protocol to satisfy certain application requirements on the reliability, efficiency, or latency of communications. It does so without the need of expert knowledge, and by minimizing the number of testbed trials executed to gauge the protocol's performance as a function of different parameter combinations. We have created a preliminary implementation of the framework coupled with the D-Cube testbed, and used it to parametrize the *Baloo-Crystal* protocol for different application requirements. Our results show that APEX can find an optimal parameter set with as few as 13 testbed trials in the median case, and reduce the average experimentation time by 65% compared to approaches based on exhaustive search.

1 Motivation

Low-power wireless (LPW) systems are becoming an integral part of the IoT and lay the foundations for a broad range of applications that are of pivotal societal importance, including smart and efficient buildings, precision agriculture, asset tracking and smart manufacturing. As there is no *"one-size-fits-all"* solution, numerous LPW communication technologies and protocols have been proposed, which allows to customize (to the extreme) a system to the application at hand, thereby maximizing its efficiency and performance [1].

Challenges. Picking the right networking stack and configuring the chosen protocols for a specific application, however, is a complex and tedious task, which requires expert knowledge and the use of adequate tools enabling a quantitative and fair performance comparison, ideally based on real-world trials prior deployment (e.g., on public testbeds).

Expert knowledge required. In fact, tuning a protocol requires significant expertise and a deep understanding of its internals, which is time-consuming to acquire. Also, it can be a very difficult exercise, particularly for individuals lacking a development background who approach the protocol solely from an application perspective. Moreover, dealing with multiple parameters simultaneously adds complexity to the parametrization process: predicting the joint behavior of multiple parameters can be challenging even for experts.

Experimentation is expensive. Configuring a system after its deployment can be challenging and expensive. Simulation frameworks and testbed facilities provide a valid alternative to field trials, enabling the optimization of a system prior deployment. Especially testbed experimentation is attractive, as it allows to test performance on actual hardware (HW) and to capture the vagaries of real-world environments, unlike simulation platforms. Unfortunately, testbeds are a precious resource shared by several practitioners, and their use is often constrained (e.g., the available runtime per user is limited on public testbeds), which makes an exhaustive testing of all possible combinations of parameter values unfeasible [2].

Lack of adequate tools. Unfortunately, to date, the community still lacks a simple tool allowing also non-experts to assess the suitability of different solutions and to automatically parametrize networking protocols based on the requirements of a given application. In fact, existing work focusing on the parametrization of LPW protocols has proposed frameworks that involve an excessive testbed use [2], a deep understanding of the protocol internals [7], or require the user to specify models of the environment and the employed HW [5].

Our contribution. We fill this gap and move the first step towards the design of APEX, an automated parameter exploration framework for LPW protocols. APEX can find an optimized parameter set for a given protocol and specific application requirements – all within a given number of testbed trials and without requiring expert knowledge. We keep APEX's design modular, such that the framework can be applied to different protocols, application requirements, and network topologies. After illustrating the main idea behind APEX (§ 2), we present a first evaluation of its functionality (§ 3), and outline our plans for its future development (§ 4).

2 APEX: Overview

The high-level architecture of APEX is illustrated in Fig. 1. Initially, the user provides key information for APEX's opti-

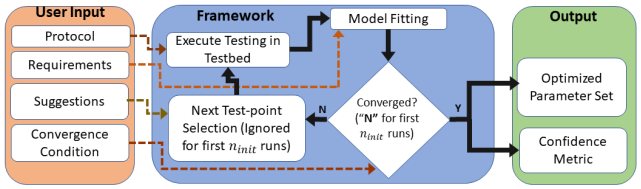

Figure 1: Overview of APEX, with its inputs and outputs. n_{init} represents the number of initial testbed trials.

mization process[1], (such as protocol under test and exposed parameters), suggestions with respect to the parameter space (such as valid range of values for a given parameter), application requirements[2], and convergence condition (CC). The latter specifies when the iterative process looking for a better parameter set terminates, and can correspond, for example, to the maximum number of available testbed trials.

The framework starts with some initial tests to gain preliminary insights into the protocol's behavior. Models are then fitted to the application requirement's goal function and constraints. Subsequently, the convergence condition is checked: if the condition is not met, the next testpoint selection (NTS) algorithm selects the next parameter set to execute in the testbed[3]. This process runs iteratively until the CC is met. Upon convergence, APEX outputs: (i) an optimized parameter set aligning with the specified requirements, and (ii) a confidence metric quantifying the level of trust in the obtained solution in terms of given constraints [4].

3 Preliminary Results

We have created a preliminary implementation of APEX and coupled it with the D-Cube testbed by leveraging its binary patching capabilities [6]. We have kept the design of APEX modular, and used linear regression for model fitting, whereas we have taken the best parameter set as per the current fitted model for the NTS (greedy for optimum).

We select *Baloo-Crystal* [3] as protocol under test (due to its deterministic behavior), and pick the transmission power and the number of re-transmissions as configurable parameters. We define requirements AR_1 and AR_2 as specified in § 2[2], and assess the performance of APEX by using D-Cube to record the results for all feasible combinations of the selected parameters (brute force). Each parameter combination is repeated ten times to incorporate variability into the analysis. Then, we run the framework over the recorded results: given the number of testbed trials as CC, we compare the performance of the parameter set returned by APEX after the given number of testbed trials with the global optimum obtained with the brute-force approach. We repeat this process 1000 times to increase the statistical significance.

Fig. 2 shows an overview of the results obtained in one out of 1000 instances when using AR_1[4]. Four distinct regions can

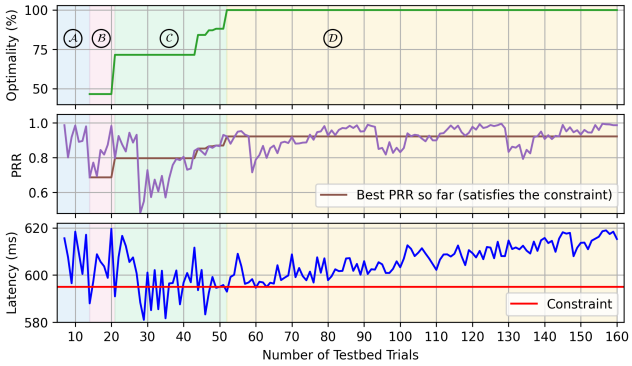

Figure 2: APEX in action parametrizing the *Baloo-Crystal* protocol when considering application requirements AR_1[4].

be observed. Ⓐ: No solution satisfying the given constraints was found. Ⓑ: A first solution satisfying the constraints was found. Ⓒ: Better solutions than the first one found were obtained, but are not yet optimal. Ⓓ: The optimal solution was found after 52 runs. When evaluating the performance of APEX over all 1000 experiments, we have observed that the framework converges to the optimum solution in as few as 16 and 10 tests (median case) for AR_1 and AR_2, respectively. In 95% of the cases, APEX converges to the optimum solution within 60 and 52 tests for AR_1 and AR_2, which represent 62.5% and 67.5% fewer tests compared to an exhaustive search. APEX outputs a confidence metric of 94.4% and 99.9% for AR_1 and AR_2, respectively: this was computed as the probability that the median satisfies the constraint.

4 Outlook

We plan to extend APEX by exploring various model-fitting approaches (e.g., Bayesian optimization with Gaussian regression) and NTS algorithms capable of incorporating uncertainty into the optimization process. We will also develop a more robust confidence metric quantifying the optimality of the provided parameter set and evaluate the framework's performance with less deterministic protocols (e.g., RPL).

Acknowledgements

This research was funded by the Austrian Science Fund (FWF) within the DENISE doctoral school (grant number DFH 5). For the purpose of open access, the authors have applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

5 References

[1] F. Foukalas et al. Dependable Wireless Industrial IoT Networks: Recent Advances and Open Challenges. In *Proc. of the 24th IEEE European Test Symp.*, 2019.

[2] S. Fu et al. Experimental Study for Multi-layer Parameter Configuration of WSN Links. In *Proc. of the 39th ICDCS Conf.*, 2022.

[3] R. Jacob et al. Synchronous Transmissions Made Easy: Design Your Network Stack with Baloo. In *Proc. of the 16th EWSN Conf.*, 2019.

[4] R. Jacob et al. Designing Replicable Networking Experiments With Triscale. *Journal of Systems Research*, 1(1), 2021.

[5] F. Oppermann et al. Automatic Protocol Configuration for Dependable IoT Applications. In *Proc. of the 10th SenseApp Workshop*, 2015.

[6] M. Schuß et al. Moving beyond Competitions: Extending D-Cube to Seamlessly Benchmark Low-Power Wireless Systems. In *Proc. of the 1st CPSBench Workshop*, 2018.

[7] M. Zimmerling et al. pTUNES: Runtime Parameter Adaptation for Low-power MAC Protocols. In *Proc. of the 11th IPSN Conf.*, 2012.

[1] We envision the user input to be provided through a YAML file that is parsed by a Python script.

[2] Each application requirement entails one/more constraints (e.g., minimum end-to-end latency) and one goal function (e.g., maximize the packet reception rate (PRR)). We refer in this work to the exemplary application requirements AR_1 and AR_2. AR_1: maximize PRR given a maximum end-to-end latency of 595 ms. AR_2: minimize energy consumption given a minimum PRR of 95%.

[3] Testbed experimentation consists in uploading and running the firmware with the modified parameters (each run lasts 10 minutes, in which 48 nodes exchange ≈ 2350 packets). Modern testbeds such as D-Cube allow to easily modify the protocol parameters of a firmware using binary patching [6].

[4] The bottom and middle plots show the avg. latency and PRR sustained by the protocol across various testbed trials during which different parameter sets are explored. The top plot shows the optimality of the solution based on the brute-forcing results (0% and 100% represent the worst and best possible solution satisfying the constraint).