To appear in: Proceedings of the 23rd Real-Time Embedded Technology and Applications Symposium, Demo Session (RTAS Demo '17)
Pittsburgh, USA, 18–21 April 2017

Demo Abstract: Tooling Support for Benchmarking Timing Analysis

Christian Eichler, Peter Wägemann, Tobias Distler, and Wolfgang Schröder-Preikschat
Friedrich-Alexander University Erlangen-Nürnberg (FAU)

***Abstract*—Precisely evaluating the accuracy of worst-case execution time (WCET) analysis tools through benchmarking is inherently difficult and in general involves a significant amount of manual intervention. In this paper, we address this problem with ALADDIN, a tooling framework that enables fully-automated evaluations of WCET analyzers. To provide comprehensive results based on benchmarks with known WCETs, ALADDIN incorporates the GENE benchmark generator. Our demonstration shows how ALADDIN evaluates two state-of-the-art WCET analyzers: the commercial tool aiT and the open-source tool PLATIN.**

I. PROBLEM STATEMENT

Ensuring timeliness is an important factor in real-time systems and requires knowledge about the worst-case execution time (WCET) of programs. The naive approach to obtain such information is to determine the maximum of the individual execution times associated with all possible program inputs, which is usually not feasible due to the wide range of input values. For example, for an input of n bits, the program had to be executed 2^n times. Existing WCET tools reduce the time required for analysis by deriving the WCET of a program from its structure. However, due to the fact that in most cases an enumeration of all possible program paths is infeasible, WCET tools resort to less extensive analyses at the cost of reduced accuracy. As a result, the values reported by analyzers in general represent overestimations of the actual WCET.

In this paper and the associated demo, we address two major problems that arise when evaluating the accuracy of WCET tools: the lack of baselines and the absence of automation.

Problem 1: Missing Baselines. The common practice to estimate the accuracy of WCET analyzers is to evaluate them with WCET benchmark suites. Unfortunately, existing suites have the drawback that they neither state the actual WCETs of benchmarks nor provide a practical way to determine them [1]. As a result, there are no baselines that would allow a comprehensive evaluation of the accuracy of WCET tools.

Problem 2: Absence of Automation. For a proper assessment of the improvements made in the field of timing analysis, the evaluation of WCET analyzers needs to be repeated periodically [2]. However, most evaluations required a significant amount of labor-intensive intervention, reducing the frequency with which evaluations can be conducted. Thus, automation of both the execution of analysis tools and the evaluation of their results is a key factor to enable frequent reevaluations.

Acknowledgments: This work is supported by the German Research Foundation (DFG), in part by Research Grant no. SCHR 603/9-2, no. SCHR 603/13-1, and the SFB/Transregio 89 "Invasive Computing" (Project C1).

II. THE GENE BENCHMARK GENERATOR

To solve the challenge of missing baselines, we developed the GENE benchmark generator that is able to create benchmarks with known WCETs [1]. To achieve this, GENE relies on small program building blocks that yield a well-known behavior. During the benchmark generation process, GENE composes these blocks in a way such that the execution of the longest path through the benchmark is triggered by a predefined input value. That is, in contrast to common WCET benchmark suites, GENE's benchmarks come with the information required to determine their actual WCET: the worst-case input value. As a result of this approach, GENE is able to provide an accurate baseline for further evaluations of WCET analysis tools. The time required for executing the longest path through the benchmark is measured by execution on the actual target hardware using the worst-case input value. The actual measurement is conducted by the ALADDIN tooling framework that is presented in the following section.

III. THE ALADDIN TOOLING FRAMEWORK

To eliminate the need for manual intervention when benchmarking timing analysis, the ALADDIN tooling framework automates the entire process of generating benchmarks, executing them on the target hardware, running WCET analyzers, and performing the actual evaluation of the values gathered in this process. For this purpose, ALADDIN incorporates reusable connectors to several components required to conduct a thorough and comprehensive evaluation of both the generated benchmarks as well as the integrated WCET analyzers, currently aiT [3] and PLATIN [4]. Even though the ALADDIN tooling framework already comes with various features, it is designed to remain flexible and thus can be extended by additional WCET analyzers or different hardware platforms.

Figure 1 illustrates the components used for the evaluation of aiT and PLATIN. The evaluation of a WCET analyzer using ALADDIN is subdivided into two different phases: During the analysis/measurement phase (illustrated in the left part of Figure 1), ALADDIN conducts all actions required to obtain raw data, such as execution-time measurements on the target hardware or upper bounds from WCET-analysis tools. The raw data generated within this phase is stored to disk for subsequent evaluations. During the evaluation phase (illustrated in the right part of Figure 1), ALADDIN reads the data of one or more benchmarks and conducts evaluations based on these datasets, such as calculating the average level of overestimation for a particular WCET analyzer.

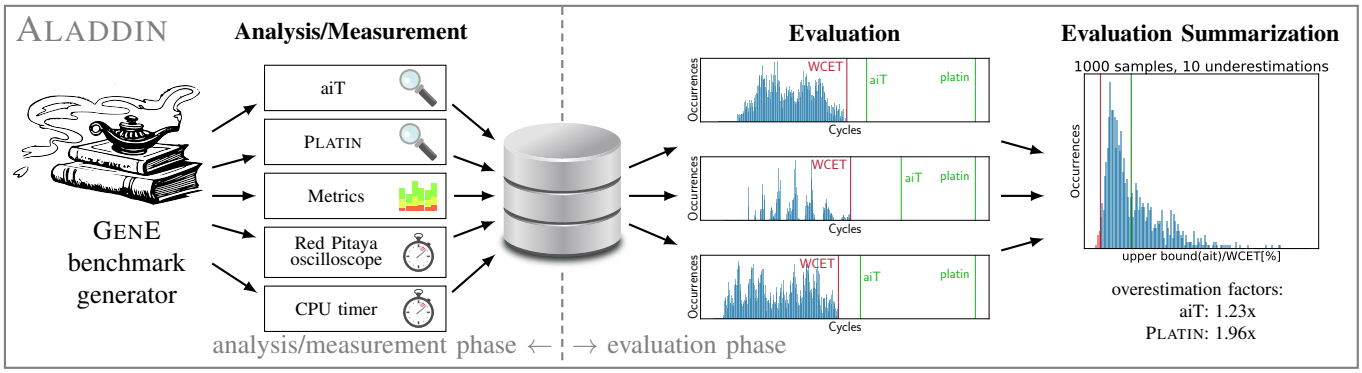

Figure 1: Application scenario of the GENE benchmark generator and the supporting ALADDIN framework

The analysis/measurement phase of ALADDIN operates on a benchmark and its worst-case input value, which are both provided by GENE. Using this input, ALADDIN determines the benchmark's actual WCET for a particular target architecture by deploying the program on the target hardware and measuring the execution time of the worst-case path. To offer further insight into the benchmark's timing behavior, ALADDIN also carries out additional measurement-based analyses by tracing the program with a large number of different input values.

ALADDIN relies on two approaches to measure execution times of program code: By default, ALADDIN uses internal CPU timers. Furthermore, ALADDIN integrates external measurement devices (i.e., oscilloscopes, logic analyzers) in order to cross-check the results provided by the internal CPU timers.

In addition to the time measurements and static analyses with aiT and PLATIN, ALADDIN applies several code metrics for WCET analysis [5] on the generated benchmark. These measures allow a classification of the analyzed benchmark and its complexity for WCET analysis.

In the evaluation phase, ALADDIN processes the data obtained in the analysis/measurement phase. This includes a) the presentation of both trace-based measurements and static analyses in a single plot and b) the summarization results derived from the evaluations of several benchmarks by different approaches. This information enables users to identify the individual strengths and weaknesses of WCET analyzers, including, for example, their level of overestimation compared to the actual WCET on a specific hardware platform.

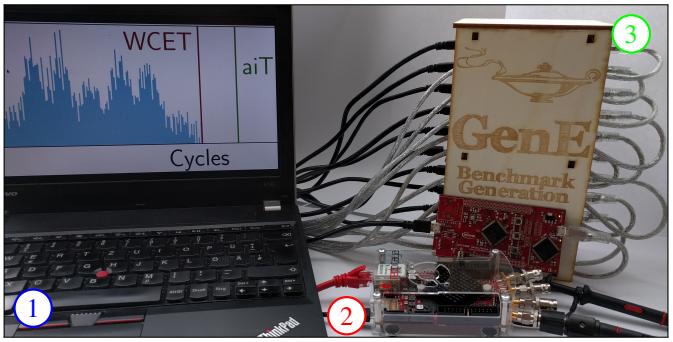

Figure 2: Demonstration setup for evaluation of WCET analyzers: (1) laptop (2) oscilloscope (3) rack (10x ARM Cortex-M4)

IV. DEMONSTRATION

In our demonstration, we present ALADDIN's abilities to determine the accuracies of two state-of-the-art WCET analyzers aiT and PLATIN (see Figure 1). This evaluation includes execution-time measurements (with CPU timers and an oscilloscope), automated WCET analyses, and the application of several code metrics. Finally, we compare and visualize the results to determine the accuracy of WCET tools.

Figure 2 shows the hardware setup required for this application scenario: The setup consists of 10 ARM Cortex-M4 boards (Infineon XMC4500), stacked in a wooden rack. All boards are connected to a laptop via USB. In addition, a remote-controllable oscilloscope (Red Pitaya) is integrated into our setup for external time measurements.

In the first step of the demo, we illustrate the step-by-step benchmark-generation process with GENE in an interactive way. ALADDIN determines the actual WCET on the target platform and executes the benchmark with different inputs. To speed up the measurements, they are distributed across all connected boards. The process of measured execution times and their occurrences is successively shown in a histogram on the laptop. In parallel to these measurements, the generated benchmark is statically analyzed by aiT and PLATIN to yield upper bounds. The code metrics, indicating the complexity for WCET analysis, are displayed on the screen.

Finally, the histogram contains all execution traces, the actual WCET, and the upper bounds from a single benchmark. We demonstrate how to automatically summarize results from several benchmarks and thereby how ALADDIN is able to reveal individual strengths and weaknesses of the WCET tools. The source code of ALADDIN, GENE [1], and the WCET metrics [5] is available at `gitlab.cs.fau.de/gene`.

REFERENCES

[1] P. Wägemann, T. Distler, C. Eichler, and W. Schröder-Preikschat, "Benchmark generation for timing analysis," in *Proc. of RTAS '17*, 2017.
[2] C. Rochange, "WCET tool challenge 2014," Talk held at WCET '14.
[3] AbsInt. aiT WCET analyzers. https://www.absint.com/ait/.
[4] P. Puschner, D. Prokesch, B. Huber, J. Knoop, S. Hepp, and G. Gebhard, "The T-CREST approach of compiler and WCET-analysis integration," in *Proc. of SEUS '13*, 2013.
[5] P. Wägemann, T. Distler, P. Raffeck, and W. Schröder-Preikschat, "Towards code metrics for benchmarking timing analysis," in *Proc. of RTSS WiP '16*, 2016.