# Supporting Swift Reaction: Automatically Uncovering Performance Problems by Systematic Experiments

Alexander Wert, Jens Happe, Lucia Happe

Karlsruhe Institute of Technology
Software Design and Quality
Am Fasanengarten 5
Karlsruhe, Germany
alexander.wert@kit.edu
jens.happe@kit.edu
lucia.kapova@kit.edu

**Abstract:** Performance problems pose a significant risk to software vendors. If left undetected, they can lead to lost customers, increased operational costs, and damaged reputation. Despite all efforts, software engineers cannot fully prevent performance problems being introduced into an application. Detecting and resolving such problems as early as possible with minimal effort is still an open challenge in software performance engineering. In this paper, we present a novel approach for Performance Problem Diagnostics (PPD) that systematically searches for well-known performance problems (also called performance antipatterns) within an application. PPD automatically isolates the problems root cause, hence facilitating problem solving. We applied PPD to a well established transactional web e-Commerce benchmark (TPC-W) in two deployment scenarios. PPD automatically identified four performance problems in the benchmark implementation and its deployment environment. By fixing the problems, we increased the maximum throughput of the benchmark from 1800 requests per second to more than 3500.

## 1 Automated Performance Problem Diagnostics

In the paper [WHH13], we introduce a novel Performance Problem Diagnostics (PPD) that automatically identifies performance problems in an application and diagnoses their root causes. Once software engineers specified a usage profile for their application and setup a test system, PPD can automatically search for known performance problems. Since PPD encapsulates knowledge about typical performance problems, only little performance engineering expertise is required for its usage. PPD combines search techniques that narrow down the scope of the problem based on a decision tree with systematic experiments. The combination of both allows efficiently uncovering performance problems and their root causes that are otherwise hard to tackle. In its current state, PPD is tailored for the diagnosis of performance problems in Java-based three-tier enterprise applications. Overall, we make the following contributions:
1) We introduce a novel approach for performance problem detection and root cause anal-

ysis called Performance Problem Diagnostics. PPD systematically searches for known performance problems in three- tier enterprise applications. Once a problem has been found, PPD isolates its root causes as far as possible.
2) We structure a large set of known performance problems in a novel Performance Problem Hierarchy. To guide PPDs search, the hierarchy starts from very general problems (or symptoms). Each further level refines the problems down to root causes. The hierarchy allows systematically excluding classes of problems and focusing on the most relevant ones.
3) We define detection strategies for twelve performance problems in the hierarchy. The strategies are based on goal-oriented experiments tailored to trigger a specific problem. Based on the results, heuristics can decide if a problem is assumed to be present and refine the search. For each performance problem, we investigated and compared different heuristics for detecting the problems. We chose those heuristics that minimize false positives and false negatives.
4) We evaluated our approach in two steps. First, we determined the detection strategies that are most likely to find a performance problem. For this purpose, we evaluated the accuracy of each detection strategy based on ten reference scenarios. Each scenario contains different performance problems which have been injected into a test application. Second, we evaluated if PPD can detect performance problems in real enterprise applications.

## 2 Summary

We used the TPC-W Benchmark for evaluation of our approach. TPC-W is an official benchmark to measure the performance of web servers and databases. Thus, we expect it to be tailored for high performance. Finding performance problems there (if any) is especially challenging (and interesting). PPD identified four performance problems and isolated their root cause by systematically narrowing down the search space. In the initial setup, the size of the database connection pool, its default implementation, the network bandwidth, and the storage engine of the database limit the maximal throughput to 1800 req/s. Solving these problems increased TPC-Ws maximal throughput to more than 3500 req/s. Based on these promising results, we can state that our approach can diagnose performance problems in real applications and detect their root cause. PPD allows software engineers to automatically search for performance problems in an application with relatively low effort. Lowering the burden of performance validation enables more regular and more sophisticated analyses. Performance validation can be executed early and on a regular basis, for example, in combination with continuous integration tests.

## References

[WHH13] Alexander Wert, Jens Happe, and Lucia Happe. Supporting swift reaction: automatically uncovering performance problems by systematic experiments. In *Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, pages 552–561, Piscataway, NJ, USA, 2013. IEEE Press.