Accurate Modeling of Performance Histories for Evolving Software Systems

Stefan Mühlbauer[1], Sven Apel[2], Norbert Siegmund[3]

Abstract: Learning from the history of a software system's performance behavior does not only help discovering and locating performance bugs, but also supports identifying evolutionary performance patterns and general trends. Exhaustive regression testing is usually impractical, because rigorous performance benchmarking requires executing realistic workloads per revision, resulting in large execution times. We devise a novel active revision sampling approach that aims at tracking and understanding a system's performance history by approximating the performance behavior of a software system across all of its revisions. In short, we iteratively sample and measure the performance of specific revisions to learn a performance-evolution model. We select revisions based on how uncertainty our models predicts their correspondent performance values. Technically, we use Gaussian Process models that not only estimates performance for each revision, but also provides an uncertainty value alongside. This way, we iteratively improve our model with only few measurements. Our evaluation with six real-world configurable software system demonstrates that Gaussian Process models are able to accurately estimate the performance-evolution histories with only few measurements and to reveal interesting behaviors and trends, such as change points.

Keywords: Software Performance; Software Evolution; Test Prioritization

Summary

Throughout a software system‘s development history, its non-functional properties, such as performance, evolve alongside. Individual modifications of the code base (*revisions*) or batches thereof can entail changes in performance. Unless identified and addressed, detrimental performance changes can add up to performance degrading over time. The retrospective analysis of existing histories can unveil causative revisions and, subsequently, help prioritize revisions for future performance regression testing. As performance measurements come at a considerable cost, it is intractable to assess all revisions. Instead, the challenge is to find a trade-off between measurement effort and accuracy of estimating performance.

We devise a novel probabilistic *active learning* algorithm to *accurately* approximate the performance history of a software system based on measurements of a specific workload *with few measurements* [MAS19]. Our approach is not only able to provide performance

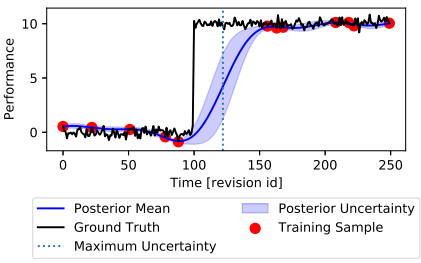
[1] Leipzig University, Institute of Computer Science, muehlbauer@informatik.uni-leipzig.de
[2] Saarland University, Saarland Informatics Campus, apel@cs.uni-saarland.de
[3] Leipzig University, Institute of Computer Science, norbert.siegmund@uni-leipzig.de

doi:10.18420/SE2021_28

estimations for all revisions, but also reports an uncertainty measure alongside. We use this uncertainty measure to decide for each revision whether our estimation is sufficiently accurate or whether we need to refine the approximation by including more measurements. To increase reliability where necessary, the algorithm selects and prioritizes new revisions for performance measurement based on the reported uncertainty and relearns the underlying Gaussian Process model.

We use *Gaussian Processes* (GPs) for time series data as a framework to model the performance history of a software system and obtain respective estimations. In a nutshell, a GP can be conceived as a distribution over functions (here: performance as a function of revisions). Evaluating the GP for a revision will yield a Gaussian $\mathcal{N}(\mu, \sigma)$ with a mean performance estimate μ and a variance measure σ. The variance σ is lower around revisions for which we have actual performance measurements at hand and can be interpreted as a measure of prediction uncertainty. The shape of an approximated performance history is determined by the GP's covariance function – a hyper parameter often called *kernel*. The kernel encodes further properties of the modeled performance histories, such as whether to expect a continuous estimation. At large, we evaluate the GP for all revisions to obtain an approximation as in Fig. 1 with regions of low and high uncertainty, the latter indicates the need for further measurements. The key idea of our approach is the following: We let the uncertainty measures *guide* the selection of new revisions to measure performance for. That is, we interpret the prediction uncertainty as a measure of how much we expect this measurement to improve the overall prediction accuracy. We repeatedly estimate performance across all revisions and add new measurements until the minimum uncertainty falls below a user-specified threshold.

Fig. 1: GP approximation of a performance history with a single change point (at revision 120).



We perform a series of experiments with the six real-world subject system from a variety of domains (file compression, scientific computing, image processing). Across different kernels evaluated, we obtained the most accurate approximations of performance histories in setups with the Brownian kernel. From such approximations, we are able to identify and pinpoint change points to individual revisions.

Bibliography

[MAS19] Mühlbauer, Stefan; Apel, Sven; Siegmund, Norbert: Accurate Modeling of Performance Histories for Evolving Software Systems. In: Proceedings of the 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, pp. 640–652, 2019.