Probabilistic Model Checking for Quantitative Evaluation of Evolving Software

Sinem Getir[1] and Lars Grunske[1]

Software Engineering, Humboldt University Berlin, Germany,

Abstract. Quantitative attributes such as reliability, safety and performance are important in the design and development of software systems. Verification of quantitative attributes via probabilistic QoS models has been a challenge for many years. The difficulty increases rapidly when the software evolves and adapt in order to comply with the requirements. Currently, quality requirements are typically checked at design time. For evolving systems with changing environmental conditions this leads to the problem that the system may behave differently with respect to quality attributes than analysed at design time. Re-verifying of an evolving system multiple times is expensive at runtime. Recently, incremental approaches have been found to be promising for software evolution and self-* systems. However, substantial improvements have not yet been achieved for evaluating structural changes in the model. In this paper, an incremental probabilistic verification framework is discussed to allow efficient evaluation of the changing models. This is complemented by a framework that supports incremental transformations and evaluation of quality models.

1 Introduction

Today's software intensive systems have to continually evolve in order to remain serviceable in the long run. Model based quality evaluation techniques are common and measurable approaches to maintain software systems consistent.

Verifying quality of service (QoS) attributes such as reliability, and performance is usually performed via probabilistic models . Modern software alters continuously and adapts to changes to improve software quality. Re-analysis of multiple versions of models is required when the system evolves and its environment changes over time. In particular, actions taken especially at run-time are often time critical. On the contrary, because of the complexity of the models, a complete re-evaluation of a quality model is time consuming. If the software engineers are considering multiple alternatives for possible system changes that need to be verified, for quality evaluations will be a bottleneck in the decision-making process. Consequently, new specific solutions are needed to reduce the time required for quality re-evaluation after a change occurs.

In this work, we are intordocing an approach for the transformation of the probablistic models into stochastic regular expressions to handle change identifactions for the evolving models.

2 Compositional and Incremental Model Based Probabilistic Verification

To achieve new incremental methods on probabilistic models, a good understanding of the models is required. In other words, the definition of incremental verification procedures requires definition of the model structures and their possible changes. Probabilistic models (e.g. Markov chains) can have two type of changes: structural and parametric changes . Compared to the overall complexity of the quality evaluation models, these changes affect only a part of the model (system). However, most of the-state-art verification approaches requires re-analyzing the entire system.This paper discusses a strategy for the incremental verification of broad variety of probabilistic models.

Firstly we propose a strategy that can speed up quality evaluation during the software evolution. This simplifies the process by determining the changes only to the part of the model in which changes occur, and propagates the result to compute the probabilistic analysis. Such methods are called delta evaluation techniques and can be considered an incremental verification procedure. To be able to apply individual techniques to the probabilistic quality evaluation models, we will generalize the methods based on promising models such as stochastic regular expressions. The reason for that, regular expressions are formally equal to an automata such that we can represent a discrete time markov chain as a regular expression with occurence distribution on regular expression. We transform (e.g. state elimination) the discrete markov chains into stochastic regular expressions as shown with an example in Fig 1. We use a syntax driven approach to get the benefit of stochastic regular expressions on a generic incremental verification tool Sidecar [2]. This technique are providing promising results to identify changed part in the syntax tree and propagate the verification metrics by using attribute grammars. Therefore, we ensure the global properties are verified by partial evaluation in the syntax tree.

Secondly, we will integrate the incremental verification framework with the graph transformation approaches developed for the change detection of the models. Validation of this approach is planned by comparison with conventional approach and state-of-theart probabilistic model checking tools. Slow down behavior can be observed for the initial input since it is translated and constructed on a regular expressions. However the speed up is expected when the model evolves with partial changes considering its large size and high complexity.

For the first validation, we plan two comparisons. Firstly, internal comparison will show the incremental verification gain versus conventional verification technique. Secondly, external comparison with state-of-the-art probabilistic analysis tools (e.g. [3]) will evaluate the incremental verification efficiency versus external tools. It is expected that a slow down behavior can be observed for the initial input since it is translated and constructed on a generic model. However the speed up is expected when the model evolves with partial changes considering its large size and high complexity. In the worst case, the whole model is re-analyzed. This case can have two causes: Either the change is small but the incremental algorithm is not efficient that it analyzes whole model or the change is quite large

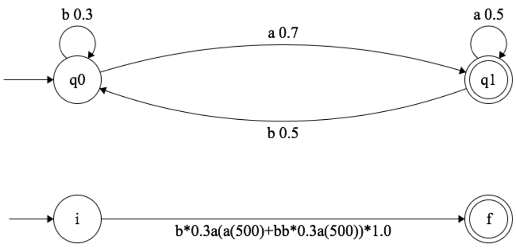

Fig. 1. Transformation of Discrete markov chains into storchastic regular expression.

as almost the model itself. Therefore, for a given change Δ on a model M and re-analyzed part Δ_R, such that $\frac{\Delta}{M} < \frac{\Delta_R}{M} < 1$, $\frac{\Delta_R}{M}$ should always approach to $\frac{\Delta}{M}$ to achieve a time efficient analysis. In other words, algorithm complexity should be polynomial with Δ ($\mathcal{O}(\Delta)$). Two possible solutions are anticipated to create an algorithm. One solution can be creating an algorithm with a new framework. The other one is creating specific algorithm for an existing framework that allows compositions of the system. In both cases, if the change is approaching the whole model, then the efficiency depends on the verification framework. Random data will be generated during the evaluation. Furthermore, common case studies in the project scope [1] will be in the evaluation plan.

For the second validation, the results will be based on the verification framework. To show the results are accurate, we plan to compare with widely used probabilistic model checking tools [3]. Accuracy tests are redundant in this approach, because the only difference may occur due to numerical approximations between the different probabilistic model checking tools.

References

1. Design for future - managed software evolution, spp 1593 case studies. http://www.dfg-spp1593.de/index.php?id=15 (2014), [Online; accessed 05-July-2014]
2. Bianculli, D., Filieri, A., Ghezzi, C., Mandrioli, D.: A syntactic-semantic approach to incremental verification. CoRR abs/1304.8034 (2013), `http://arxiv.org/abs/1304.8034`
3. Kwiatkowska, M.Z., Norman, G., Parker, D.: Prism 4.0: Verification of probabilistic real-time systems. In: CAV. pp. 585–591 (2011)