The Journal of Systems and Software 147 (2019) 1–16

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Online model learning for self-aware computing infrastructures

Simon Spinnner, Johannes Grohmann*, Simon Eismann, Samuel Kounev

University of Würzburg Am Hubland, Würzburg 97074, Germany

ARTICLE INFO

Article history:
Received 30 September 2017
Revised 31 July 2018
Accepted 28 September 2018
Available online 2 October 2018

Keywords:
Self-aware computing
Performance model
Model extraction
Model learning

ABSTRACT

Performance models are valuable and powerful tools for performance prediction. However, the creation of performance models usually requires significant manual effort. Furthermore, as the modeled structures are subject to frequent change in modern infrastructures, such performance models need to be adapted as well. We therefore propose a reference architecture for online model learning in virtualized environments, which enables the automatic extraction of the aforementioned performance models. We follow an agent-based approach, which enables us to incorporate the extraction of information about the application structure as well as the virtualization structures present in modern computing centers. Our evaluation shows that our collaborating agents are able to reduce the manual effort of performance model extraction by 85.4%. The resulting performance model is able to predict the system utilization with an absolute error of less than 4% and the end-to-end response time with a relative error of less than 21%.

1. Introduction

IT services hosted in data centers, such as public internet services (e.g., Netflix, Facebook, or Google) as well as intranet services in corporate networks, are typically subject to time-varying workloads. Changes in the current number of users or their interactions with a service influence its resource demand. At any point in time, the amount of resources allocated to an application needs to fulfill its current demand. As a result, continuous adaptations to the resource allocations of an application are required during operation.

Current approaches to automatic resource management in industry are based on a rule-based approach: A system administrator manually defines custom triggers that fire when a metric reaches a certain threshold (e.g., high resource utilization or load imbalance) and execute certain reconfiguration actions. However, application-level performance metrics, such as response time, normally exhibit a highly non-linear behavior on system load. Therefore, it is not possible to determine general thresholds of when triggers should be fired, given that the appropriate triggering points are typically highly dependent on the architecture of the hosted services and their usage profiles, which can change frequently during operation.

In order to overcome the limitations of rule-based approaches, the usage of different kinds of performance models has been proposed in the literature (Jennings and Stadler, 2015; Lorido-Botran et al., 2014). Stochastic performance models (e.g., queueing networks, or descriptive modeling languages such as DML (Huber et al., 2017)) provide powerful abstractions of a combined hardware and software system describing its performance-relevant structure and behavior. They can predict the impact of a reconfiguration action on the system performance in advance and thus promise significant improvements for the automatic resource management of IT services. Existing model-based approaches either abstract the application as a black-box severely limiting their prediction capabilities; or expect manually created model instances as input. However, the expertise and effort required to create and maintain such detailed models of the infrastructure and applications in a virtualized environment manually pose a major challenge to exploit the advanced prediction capabilities of stochastic performance models for automatic resource management.

Challenges. In this article, we describe a new agent-based reference architecture enabling the deep integration of online model learning capabilities into virtualized environments. Our reference architecture addresses the following challenges:

- Given that model learning is performed during system operation, the system workload and configuration cannot be controlled. We rely on empirical observations while applications are serving production workloads. In order to avoid significant overheads on the performance of services, existing monitoring infrastructures and platform interfaces should be used to obtain the empirical information required for model learning.
- The integration of model learning capabilities into systems requires a pro-found understanding of the system architecture – including the application and any platform layers – and at the

* Corresponding author.
E-mail addresses: simon.spinnner@uni-wuerzburg.de (S. Spinnner), johannes.grohmann@uni-wuerzburg.de (J. Grohmann), simon.eismann@uni-wuerzburg.de (S. Eismann), samuel.kounev@uni-wuerzburg.de (S. Kounev).

https://doi.org/10.1016/j.jss.2018.09.089
0164-1212/

same time a deep knowledge of performance modeling techniques. However, system administrators often do not have sufficient skills to perform such tasks. Furthermore, it can be time-consuming and costly to design and implement model learning capabilities for a given system. Therefore, ways to enable the reuse and sharing of model learning capabilities between systems are necessary.
- Multiple applications with diverse technology stacks typically share the same underlying infrastructure in virtualized environments influencing each other. A performance model needs to represent the complete virtualized system (including the different applications) integrating information from heterogeneous datasources. However, the deployment of applications and their software stacks are often not known before system run-time (especially with the advancement of on-demand provisioning of Virtual Machines (VMs) in cloud environments). As a result, the end-to-end performance model of the system can only be dynamically composed a system run-time.
- The deployment and configuration of applications may change frequently due to automatic or manual reconfigurations (e.g., deployment of new VMs, or migration of existing ones). As a result, the overall performance model of the system needs to be continuously updated to always reflect the current system architecture and configuration.

A major field of research is the automatic extraction of performance models based on static and dynamic analysis of the system implementation and configuration in order to ease the usage of performance models. Existing work either describes holistic approaches to extract complete performance models, but assume a very specific technology stack (Brosig et al., 2011; Brunnert et al., 2013), or focuses on improving certain aspects of it (e.g., resource demand estimation (Spinner et al., 2015)).

Contributions. In this paper, we propose an agent-based reference architecture for online model learning in virtualized environments. In particular, this paper makes the following contributions:

- We extend the notion of Virtual Appliances (VAs) to include model learning capabilities in order to automatically build and maintain submodels describing performance behavior of the application architecture and infrastructure layers.
- We introduce additional components into the virtualization platform to collect these sub-models and dynamically compose them into an end-to-end performance model of the complete system.
- We identify different roles an agent may take over during model learning and describe the required communication between agents in different roles.
- We develop an algorithm for merging the different model skeletons into a complete performance model in a central repository.

In order to evaluate this, we create a reference implementation of the proposed agent structure monitoring a distributed SPECjEnterprise2010 benchmark. We evaluate the degree of automation for the model extraction as well as the prediction accuracy of the resulting model. Although targeted at virtualized environments, some aspects, like the use of agent-based architectures for performance model extraction of distributed software systems and the proposed model merging algorithm, can be transferred to other application domains.

Structure. The remainder of the article is organized as follows. Section 2 introduces our proposed reference architecture for integrating model learning capabilities into virtualization platforms. Section 3 gives an overview of the related work. Section 4 describes our reference implementations of three different model learning agents. We evaluate our reference implementation in Section 5. Lastly, we summarize and conclude our work in Section 6.

2. A reference architecture for online model learning

Modern hypervisors (e.g., VMware ESX or Xen) and virtualization management software (e.g., VMware vCenter) - in the following, the combination of both is called the *virtualization platform* - rely on standardized formats for VM images to support the deployment of new VMs. However, this image format is focused on the specification of the virtual hardware resources including their configuration and lacks meta-data describing the platform and application layers inside a VM. The program code of the platform and application layers, as well as any additional data, is contained in an unstructured binary image of the virtual hard disk.

Therefore, a virtualization platform is generally not aware of what is contained inside a VM. Although, the virtualization platform may access all data in the main memory and hard disks of a VM, the data is hard to interpret given that no general assumptions can be made on their structure. An approach to model learning solely based on information available in the virtualization platform inevitably leads to performance models abstracting application and platform layers as a black-box. In contrast, an approach based on model learning inside a VM may provide detailed performance models of the platform and application layers running in the same VM. However, in the latter case access to the underlying infrastructure layers or co-located applications is prohibited. In the following, we describe our reference architecture for model learning that bridges this gap.

2.1. Conceptual overview

We argue that model learning capabilities should be integrated deeply into both the virtualization platform and the hosted VMs enabling the extraction of end-to-end performance models covering the virtual infrastructure, as well as any platform and application layers within VMs. We assume a virtualization platform that hosts a set of *VAs*. A VA is a set of pre-packaged VM images each containing a complete software stack ready to run on a virtualization platform VAs can significantly reduce the effort and knowledge required for deploying software systems. VAs are either provided directly by software companies or by individuals. VAs are built by experts of the respective system and can then be shared with others (e.g., through online marketplaces, such as VMware Solution Exchange[1]). When deploying such a VA, only certain predefined settings may need to be customized (e.g., through a web interface provided by the VA) in order to adapt it to a target virtual environment (e.g., IP address settings, or passwords).

Our reference architecture is based on an extension of conventional VAs to include additional logic for learning performance models of the application as well as any contained platform layers (e.g., middleware systems or Java VMs) during system run-time. The model learning logic is encapsulated in specialized *agents* distributed as part of a VA. On instantiation of such a VA in a virtualized environment, the contained agent will start to monitor the application serving real production workloads and will automatically build a sub-model (so-called *model skeleton*) describing the observed performance behavior of the application and platform layers inside the VA. The agent continuously updates the model skeleton to reflect dynamic changes, for instance, in the configuration or the workload of an application. A virtualization platform may access the model skeletons extracted by the agents of a VA using a defined interface in order to obtain fine-grained performance models of an application.

[1] https://solutionexchange.vmware.com/store.