Application Resource Demand Phase Analysis and Prediction in Support of Dynamic Resource Provisioning

Jian Zhang*, Mazin Yousif†, Robert Carpenter‡, and Renato J. Figueiredo*
*University of Florida, Gainesville, FL, USA
†Intel Corporation, Hillsboro, OR, USA
‡Intel Corporation, Folsom, CA, USA

Abstract

Profiling the execution phases of an application can lead to optimizing the utilization of the underlying resources. This is the thrust of this paper, which presents a novel system-level application resource demand phase analysis and prediction prototype to support on-demand resource provisioning. The phase profile learned from historical runs is used to classify and predict phase behavior using a set of algorithms based on clustering. The process takes into consideration application's resource consumption patterns, pricing schedules defined by the resource provider, and penalties associated with Service-Level Agreement (SLA) violations.

1. Introduction

In systems such as Amazon's Elastic Compute Cloud (EC2) [1], which allow users to reserve and reconfigure resource allocations and charge based upon such allocations, users have an incentive to request no more than the amount of resources an application needs. In this work, we focus on modeling and analyzing long-running applications' phase behaviors based on monitoring and learning of their historical resource consumption patterns, which may vary widely over their run time. Understanding these behaviors can unlock a variety of optimizations of resource scheduling in an autonomic system.

To enable the self-optimization of the configuration of the application's execution environments, we must first develop analytical tools necessary to automatically and efficiently discover similarities and changes in application's resource consumption behavior over time. We refer to this as "phase behavior." In this context, a *phase* is defined as a set of intervals within an application's execution that have similar system-level resource consumption behavior, regardless of temporal adjacency. *Phase classification* partitions a set of intervals into phases with similar behavior. In this paper, we introduce an application resource demand phase analysis and prediction prototype, which uses a combination of clustering and supervised learning techniques to investigate the following questions: 1) How many phases should be used to provide optimal resource provisioning? 2) What is the next phase of the application's execution?

This prototype takes the application's resource consumption patterns, phase transition costs, and misprediction penalties associated with SLA violations into account during the optimization. The prediction performance is used as a feedback signal to guide the future phase analysis. The prototype does not require any instrumentation of the application source codes and can generally work with both physical and virtual machines which can provide monitoring of system level performance metrics.

2. Application resource demand phase analysis and prediction prototype

Figure 1 illustrates our application phase analysis and prediction prototype. In this prototype, a virtual machine (VM) [2] is used as a resource container to host the application execution. It is referred to as application VM. Therefore, the application VM's system level performance data reflect and summarize the application's resource consumption [3]. The prototype models how the performance data are collected and analyzed to construct application's phase profile and how the profile is used to perform phase predictions.

The performance data are collected by a *monitoring agent* installed in the Virtual Machine Monitor (VMM) and stored in a *performance database*. The *phase analyzer* retrieves the time-series VM performance data from the database and performs phase analysis

Fourth International Conference on Autonomic Computing (ICAC'07)
0-7695-2779-5/07 $20.00 © 2007 IEEE

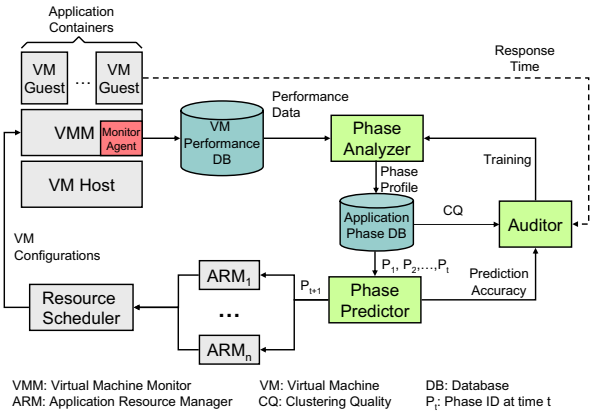

Figure 1. Application resource demand phase analysis and prediction prototype

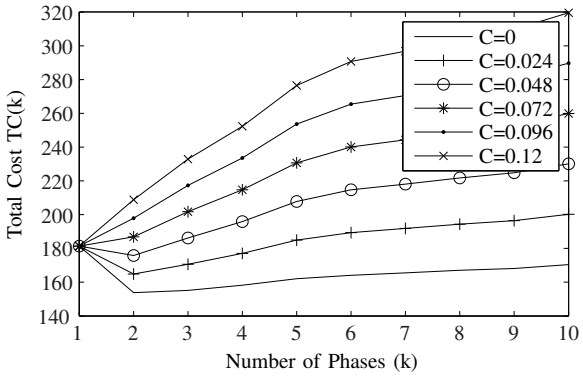

Figure 2. VM1 VCPU0 total cost with penalty
Relative Unit Cost of Misprediction Penalty $C_p = 8$

using algorithms based on clustering to find out how many phases in a numeric range are best in terms of providing the minimal total costs. The output *phase profile*, which consists of the defined number of phases, corresponding cluster centroids and resource usage statistics of each phase are stored in the *application phase database*.

The phase profile is used to train the phase predictor, which can perform on-line prediction of the next phase of the application's resource usage based on the learning of historical phase behaviors. The predicted phase information can be used to support the *application resource manager*'s decisions regarding resource re-provisioning request to the *resource scheduler*.

The *auditor* monitors and evaluates the phase analysis and prediction performance. If the performance drops below a predefined threshold, the auditor can order a phase analysis to update the phase profile with recent workload and retrain the phase predictor.

3. Empirical Evaluation

We have implemented a prototype for the phase analysis and prediction model including Perl and Shell scripts to extract and profile the performance data from the performance database, and a Matlab implementation of the phase analyzer and predictor. The total cost $TC(k)$ of a k phase workload is defined as sum of resource reservation cost $R(k)$, phase transition cost $TR(k)$, and misprediction penalties $P(k)$.

$$TC(k) = R(k) + C \times TR(k) + C_p \times P(k) \qquad (1)$$

where C and C_p denote the relative unit cost of phase transition and the relative unit cost of misprediction penalty with regard to the unit cost of resource reservation. A seven-day trace of physical CPU usage (VCPU0) of a virtual machine, which hosts a web server, Globus GRAM/MDS and GridFTP services, and a PBS head node, is used in the experiment. Figure 2 depicts the relationship between the number of phases k and the total costs $TC(k)$ for $C_p = 8$ and a set of sample values of C, starting from $C = 0$ that shows the case of no transition cost. The optimal number of phases happens when k gives the lowest cost.

4. Summary

The application resource demand phase analysis and prediction prototype presented in this paper shows how to apply statistical learning techniques to support on-demand resource provisioning. It also shows how to define the phases in the context of system-level resource provisioning and provides an approach to automatically find out the number of phases needed to provide optimal cost. The proposed cost model takes the resource cost, phase transition cost, and prediction accuracy into account. With the knowledge of system-level application phase behavior, we can perform dynamic optimization of resource scheduling to improve system utilization and to reduce the users' cost. This is part of our future research.

References

[1] Amazon Elastic Compute Cloud (Amazon EC2).

[2] J. Smith and R. Nair, *Virtual Machines*. Morgan Kaufmann, Jun. 2005.

[3] J. Zhang and R. Figueiredo, "Application classification through monitoring and learning of resource consumption patterns," in *Proc. of IPDPS'06*, Rhodes Island, Greece, Apr. 25–29, 2006.