Future Generation Computer Systems 28 (2012) 104–111

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

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

Feedback-based optimization of a private cloud

Hamoun Ghanbari [a,b,*], Bradley Simmons [a], Marin Litoiu [a], Gabriel Iszlai [b]

[a] Department of Computer Science and Engineering, York University, 4700 Keele St, Toronto, Ontario, M3J 1P3, Canada

[b] Centre for Advanced Studies (CAS), IBM Toronto Lab, 8200 Warden Avenue, Markham, Ontario, L6G 1C7, Canada

ARTICLE INFO

Article history:
Received 21 January 2011
Received in revised form
11 May 2011
Accepted 28 May 2011
Available online 6 June 2011

Keywords:
Optimization
Modeling
State estimation
Private cloud
PaaS
IaaS

ABSTRACT

The optimization problem addressed by this paper involves the allocation of resources in a private cloud such that cost to the provider is minimized (through the maximization of resource sharing) while attempting to meet all client application requirements (as specified in the SLAs). At the heart of any optimization based resource allocation algorithm, there are two models: one that relates the application level quality of service to the given set of resources and one that maps a given service level and resource consumption to profit metrics. In this paper we investigate the optimization loop in which each application's performance model is dynamically updated at runtime to adapt to the changes in the system. These changes could be perturbations in the environment that had not been included in the model. Through experimentation we show that using these tracking models in the optimization loop will result in a more accurate optimization and thus result in the generation of greater profit.

1. Introduction

Advances in virtualization [1] techniques and the construction of numerous large commodity data centers around the world [2] have resulted in a new approach to computing referred to as *cloud computing* [3,2,4,5] becoming an important topic of research and development. The *cloud*, though still in its infancy, typically refers to some set of computing resources (infrastructure (IaaS), platform (PaaS) or software (SaaS)) being provided on demand over the Internet to users as a service.

A *private cloud* represents a set of virtualized data centers under the ownership of a single administrative domain (i.e., the cloud service provider). Unlike in a *public cloud* where the various layers may be offered by multiple providers the entire stack (IaaS,PaaS and SaaS) is controlled by a single provider and so it has access and control over the various applications, middlewares and infrastructure simultaneously. The main objective of a private cloud provider is to maximize *profit* (i.e., revenue–cost). Optimization techniques allow the provider to determine resource allocations to various clients in order to best maximize its revenue while minimizing its costs.[1] Due to these economic benefits, optimization has been the subject of much investigation [11–18].

The decisions made by a provider with regards to deployment of application tiers in the cloud and resource allocation to application environments can be enforced through *scale-up/down* (i.e., adding/removing resources to individual virtual machines (VM)), *scale-out/in* (i.e., adding/removing VMs to an application environment), and *migration* (i.e., moving VMs over the physical infrastructure) and will directly impact both the performance of an application and the provider's cost of operations. Here we focus only on scale-up/down.

The optimization problem addressed by this paper involves the allocation of resources in a private cloud such that cost to the provider is minimized (through a maximization of resource sharing) while attempting to meet all client application requirements as specified in their respective Service Level Agreements (SLA)s[2] [19–22] (see Fig. 1). Many of the current optimization

* Corresponding author at: Department of Computer Science and Engineering, York University, 4700 Keele St, Toronto, Ontario, M3J 1P3, Canada. Tel.: +1 416 265 7585.

E-mail addresses: hamoun.gh@gmail.com, hamoun@cse.yorku.ca (H. Ghanbari), bsimmons@yorku.ca (B. Simmons), mlitoiu@yorku.ca (M. Litoiu), giszlai@ca.ibm.com (G. Iszlai).

URLs: http://www.ceraslabs.com/people/hamoun-ghanbari (H. Ghanbari), http://www.ceraslabs.com/people/dr-bradley-simmons (B. Simmons), http://www.ceraslabs.com/people/dr-marin-litoiu (M. Litoiu).

0167-739X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2011.05.019

[1] Notice that in a layered cloud, optimization is decomposed into a dynamic infrastructure pricing mechanism offered by provider [6,7] and elastic resource allocation policies employed by individual consumers [8–10] to satisfy their QoS requirements.

[2] An SLA is a contract which defines the relationship between a service provider and its clients that fully specifies all obligations for both parties, the price to be paid for the service(s) offered and associated penalties should obligations be unmet. It can be quite complex and comprehensive (e.g., considering aspects of both functional and non-functional requirements); however, in this work, only performance objectives that can be extracted from an SLA are considered. No attempt is made to fully model or develop an SLA or an SLA management framework.

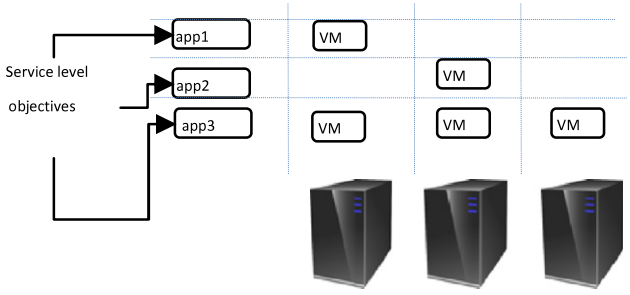

Fig. 1. The interaction of layers in our optimization mechanism.

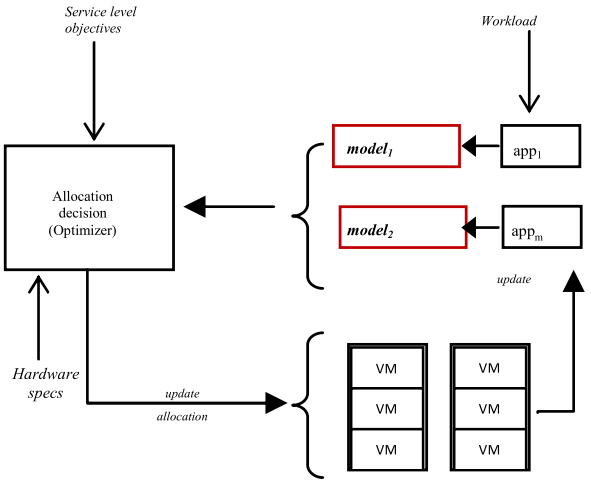

Fig. 2. Feedback based optimization of resource shares.

approaches, while efficient, assume static models [11,23,24,12–18]. In this paper we attempt to solve the optimization problem through a feedback-based loop with dynamic models. The resulting optimizer will be compared to one using static models to demonstrate the benefits of this proposed approach.

The remainder of this paper is structured as follows: Section 2 describes our feedback based approach for optimization of resource shares in a private cloud. This involves the introduction of general formal definitions used during problem formulation and the description of the estimator component of the feedback loop. Section 3 explains the optimizer component of the loop. The applicability of the proposed approach is demonstrated by the case studies in Section 4. Related work, conclusions and future works are discussed in Sections 5 and 6.

2. Proposed approach

We propose a feedback based Cloud Optimization Manager (COM) as shown in Fig. 2. In this approach, each application maintains both a dynamically updated performance model (see Section 2.3) and a utility model (which defines a specific service level objective e.g., response time). The COM has access to the utility model of each application. Periodically, each application submits its performance model to the COM which performs a system-wide global optimization (see Section 3) using this information and determines new resource allocations for each application for the following period.

Consider applications $app_1, \ldots, app_m$. Each application runs within one or more VMs and experiences a particular workload. Assume that there are n physical machines (PM)s in the data center, represented by the set $P = p_0, \ldots, p_n$. VMs are hosted on PMs on behalf of applications. Let us assume that the physical server environment is homogeneous and each physical machine,

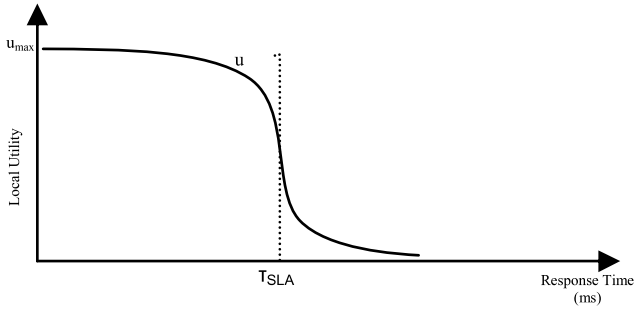

Fig. 3. Smooth service level utility function; vertical line indicates the servicel level objective of an application (as defined in SLA).

say p_i, has one resource, and thus has a fixed capacity c_i in one-dimensional space.[3] The allocation of VMs on PMs can be represented by an $n \times m$ matrix A. Each element a_{ij} of A denotes a resource allocation defining the percentage of the total resource (i.e. CPU) capacity of the PM i allocated to a running VM of application j.

2.1. Problem formulation

A global utility function U_0 is expressed as the difference of the sum of application-provided resource-level utility functions and an operating cost function as follows:

$$U_0 = \left(\sum_{j \in App} u_j(s_j(A_j))\right) - \omega.cost(A) \tag{1}$$

where ω denotes an adjustable weight (working as a tunable parameter for the administrator), s_j denotes the service level function which maps the application j's resource allocations (i.e. A_j) to the service level measure (e.g. response time) of the application, u_j is the local utility function for application j, A represents the allocation matrix of VMs on PMs (defined earlier), and A_j represents the j'th column of A. U_0 can be associated with the profit of the cloud and the two terms of Eq. (1) represent the revenue and the cost respectively.

Our objective is to maximize U_0 subject to a set of capacity constraints which come from the physical layer of the private cloud. The optimization problem addressed here can be expressed as follows:

$$\begin{aligned} &\text{maximize:} \quad U_0 \\ &\text{subject to:} \quad \forall i \left(\sum_{j \in App} a_{ij} < c_i\right) \\ &\qquad\qquad \forall i\, \forall j\ (a_{ij} \in [0, c_i]). \end{aligned} \tag{2}$$

It is assumed that each allocation signal a_{ij} is constrained to lie in the interval $[0, c_i]$ meaning that an application can get the whole capacity of a PM.

Notice that the problem has a best effort nature and we treat a service level objective (i.e. target on a specific QoS metric) as a soft constraint by incorporating it into the objective function.

Fig. 3 represents a sample service level utility function where the vertical line indicates the SLA target of an application and utility decreases as the value of s_j approaches the SLA limit.

It is worth noting that any decreasing differentiable function can be used instead. However, the shape of a function, especially after passing the SLA threshold, will impact the behavior of the optimization algorithm.

[3] In the case of multi-resource modeling c_i can be substituted with c_i^r, where c_i^r is the capacity of resource r of p_i.