Journal of Network and Computer Applications 95 (2017) 94–104

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

Affinity aware scheduling model of cluster nodes in private clouds

Daniel Yokoyama[a,*], Bruno Schulze[a], Henrique Kloh[b], Matheus Bandini[b], Vinod Rebello[b]

[a] Computer Science, National Laboratory for Scientific Computing (LNCC), Av. Getulio Vargas, 333-Quitandinha, 25651-075 Petropolis, Rio de Janeiro, Brazil

[b] Instituto de Computação – Universidade Federal Fluminense (UFF), Niterói, Rio de Janeiro, Brazil

ARTICLE INFO

Keywords:
Private clouds
Scheduling
Tasks affinities
Interference
Virtualization

ABSTRACT

Running applications on a cloud environment without checking and meeting their allocation and performance requirements may lead to unexpected application slowdown and infrastructure under-utilization. In addition, competition for same shared resources may cause performance degradation when applications with similar resource usage profiles are scheduled concurrently. This paper presents an affinity-based model for scheduling virtual machines that host and run scientific applications on a private cloud environment. The main contributions of the proposed model are: i) an approach that exploits the concept of affinity relations among competitive applications; ii) a set of experiments using consolidated HPC benchmarks and their analysis to assess the performance of two concurrent applications; and iii) a novel scheduling algorithm based on an affinity relation among competitive applications.

1. Introduction

1.1. Motivation

The increasing complexity of applications, particularly scientific applications, associated with the need to manage large amounts of data, is driving a growing demand for high performance and high distributed computing architectures, such as cluster computing, in order to obtain solutions for these problems, within acceptable time constraints. The use of both cluster computing and parallel processing leads to the simulation and solving of complex problems that would not be achieved otherwise.

However, cluster computing presents some barriers to its widespread adoption, such as the complexity of applying large scale distributed parallelism and the difficulty of accessing cluster resources, which is not trivial for scientist in general areas of interest. Cloud computing emerged as an alternative to deal with such issues, as it may reduce infrastructure maintenance costs and provide easier ways to experiment and develop parallel solutions (Evangelinos and Hill, 2008).

Because of recent developments, such as hardware assisted virtualization in x86 processors, the cloud computing model, although not new, is attracting great interest from scientific communities. Cloud computing offers an availability of computational resources in an easier way and on demand, reducing entry costs and infrastructure maintenance (Mell and Grance, 2011). This helps to mitigate some of the challenges presented in High Performance Computing (HPC).

Most existing cloud platforms depend heavily on virtualization of the computing resources. Virtualization allows for: a reduction of equipment purchase costs, by taking advantage of underutilized facilities;a greater flexibility by using the same hardware for a range of applications running possibly on different operating systems; an increased stability and environmental safety, since a failure in a virtual machine will not be propagated to other virtual machines running on the same host. Observing the listed benefits, it becomes clear why clouds depend intrinsically on virtualization (Xing and Zhan, 2012).

Applications in clusters are comprised of largely homogeneous tasks across distributed memory systems. These tasks, when isolated as virtual machine instances in a private cloud computing environment, present great opportunities to analyze their relationship with other applications submitted to the same host and to allocate them accordingly. Thus, the objective of this paper is to present an allocation model for Virtual Machines (VMs) in a private cloud infrastructure in support to scientific applications. This model aims to reduce the costs of moving cluster computing applications to cloud computing environments, as well as to mitigate negative effects that arise from the competition for the same computing resources in a virtual environment. Thus, the benefits of cloud computing, such as scalability, elasticity and resource sharing, would be exploited by a cluster computing infrastructure.

* Corresponding author.
E-mail address: dmassami@lncc.br (D. Yokoyama).

http://dx.doi.org/10.1016/j.jnca.2017.08.001
Received 2 January 2017; Received in revised form 26 June 2017; Accepted 1 August 2017
Available online 03 August 2017
1084-8045/

1.2. Methodology

Based on the analyses of the interaction of different applications with different resource constraints, and through benchmarks and validation via simulations, this work proposes a scheduling model to improve private cloud resource utilization. Currently,the scheduling mechanism used in the cloud do not take into account how applications affect the overall system utilization, due to resource competition. This work proposes a model that take this interaction into account in order to maximize application throughput.

The co-allocation effect is measured throughout the execution of benchmarks with different performance characteristics. The impact of the hypervisor is overlooked by this allocation model. Although the type of hypervisor can affect performance, cloud environments tend to use a single Virtual Machine Monitor (VMM). To this end, the KVM hypervisor is used as the VMM in all experiments. KVM was chosen since it has shown to be well suited for applications that require intensive processing, in some cases supplanting the real machine (Luszczek et al., 2012, Yokoyama et al., 2012).

Finally, simulations are performed using some traditional scheduling strategies and the proposed model based on affinities. These simulations have the objective of validating the model.

The remaining sections are: Section 2 presents the relation between interference and affinity,detailing the complexity of virtual machine instance allocation in cloud datacenters and explaining the hypothesis under witch this work was developed; Section 3 briefly explains the benchmarks used as applications and the results that ascertain the interference among virtual machines in a host; Section 4 explains how affinity is used to decide where a virtual machine instance should be allocated; Section 5 briefly explains the scheduling methods used in this work, including standard scheduling policies, and the proposed model; Section 6 makes use of simulations to verify the hypothesis proposed by this work and, based on the results, proposes an affinity aware scheduling model; Section 7 presents a review of related works that deal with virtual machine scheduling and interference; Section 8 summarizes the results achieved by this work and proposes future developments that could lead toa better use of cloud resources.

2. Problem specification

In the context of this work, a cluster is a set of virtual machines instantiated at the time of execution of a specific application. These virtual machines are dedicated to solve a single distributed memory parallel job. The evaluated clusters use Message Passing Interface (MPI) in a distributed memory environment.

Traditionally, a job represents the entire computational work that has to be processed by a cluster. However, in the context of this work, the term "job" is interchangeable with cluster in execution, i.e., the proposed model does not schedule jobs, but the entire system (virtual machines) that contains the said jobs. In other words, a job is composed of all the virtual machines loaded within the process to be executed. The term "task" refers to a job processing unit, therefore, task refers to the number of MPI running tasks. The term "instance" refers to each virtual cluster node created in the cloud computing environment.

It is known that the total processing capacity of a computing system may vary greatly due to the interference of the applications running on the same host (Mury et al., 2015), the type of hypervisor (as it may be more suitable for one type of application, while another type may present significant losses due to virtualization overhead), and so on. So, the total processing capacity may be reduced, depending on how the problems were allocated. Thus, the main focus of the proposed model is to find the best application combinations to reduce interference among tasks. Two applications that have fewer interference between them, due to reduced impact of competition for resources in a host, are henceforth called"affine". Therefore affinities, in the context of this work, are normalized values of application performance when executed concurrently. An affinity of 1 represents two jobs whose competition does not result in any negative effects in performance, i.e., zero interference. An affinity of 0 represents jobs that cannot be completed because of their competition. The affinity of n concurrent jobs is obtained, in this work, as the arithmetic means of a performance parameter of n jobs in parallel in respect to the same jobs when running isolated. Eq. (1) expresses the affinity of n concurrent jobs ($A_{j1,j2,\ldots,jn}$), where $P_{j1,j2,j3,\ldots,jn}$ is a measurement (time (t^{-1}), flops, etc.) of job 1 executing in parallel with the other n jobs.

$$A_{j1,j2,\ldots,jn} = \frac{\frac{P_{j1,j2,j3,\ldots,jn}}{P_{j1}} + \frac{P_{j2,j1,j3,\ldots,jn}}{P_{j2}} + \cdots + \frac{P_{jn,j1,j2,\ldots,jn-1}}{P_{jn}}}{n} \quad (1)$$

The term affinity used in this work first appears in the work (Licht, 2014). To the authors' knowledge, Licht (2014) is the first time this term was used in this context. This term is employed in this work to denote tasks which cooperate better in a co-allocated scenario.

2.1. Problem analysis

To better understand the contribution of this work, it is helpful to analyze the complexity of allocating jobs among many hosts. The problem can be summarized as: solving how to allocate a number of instances I on H hosts, each one capable of hosting at most l_i instances. Assuming that each host can receive from 0 to I instances, the analyzed problem is a weak composition. A weak composition allows for the inclusion of the identity(0). The composition of a positive integer s is given by the list consisting of all positive integers whose sums results in s. Thus, for example, let $s = 3$ $C_3 = 1 + 1 + 1;\ 1 + 2;\ 2 + 1;\ 3$, where C_3 is the list of the composition of the number 3. The number of parts of the list of the composition of s is called length of the composition(n). Weak composition includes the digit 0, so the list is unbounded, adding zeros to the end of the sum. By limiting the number of digits we have a problem that better resembles the one treated in this work. Page (2013) presented the following definition: let $n \in \mathbb{Z}^+$ and $s \in \mathbb{Z}^+ \bigcup \{0\}$, the weak composition $C_{s,n}$ is the set of any non-negative integer sequences $\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$, where $\sigma_i \in \mathbb{Z}^+ \bigcup \{0\}$, and $\sum_{i=0}^{n-1} \sigma_i = s$. From Reingold et al. (1977), the cardinality of $|C_{s,n}| = \binom{n+s-1}{n-1}$.

This abstraction of the allocation problem allows to the analyses of the maximum range of the addressed problem. Based on the work described in Page (2013), we assign restrictions on possible values of the parts of the sum. Let $n \in \mathbb{Z}^+$, $s \in \mathbb{Z}^+ \bigcup \{0\}$ and the restricted set R^1, such that $R^1 \in \mathbb{Z}^+ \bigcup \{0\}$ e $0 \le R^1 \le s$. The first-order restricted weak composition $C_{s,n}^{(R^1)^n}$ is the set of sequences of any positive integer $\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$, where $\sigma_i \in R^1$, and $\sum_{i=0}^{n-1} \sigma_i = s$. As an example, given the restriction $0 \le R^1 \le 2$:

$$C_{3,3}^{(R^1)^3} = \{(1, 1, 1); (1, 2, 0); (1, 0, 2); (0, 1, 2); (0, 2, 1); (2, 1, 0); (2, 0, 1)\} \quad (2)$$

This definition differs from that presented in Page (2013). In the referenced work we have $R^1 \subseteq \{0, 1, \ldots, s\}$. For the problem addressed in this paper, there is not a host capable of supporting two instances, for example, which is not capable of supporting only one instance. That is if H has $l_i = n \Rightarrow$, H accepts $I = \{n, n-1, n-2, \ldots, 0\}$

This improved abstraction still does not perfectly fit the problem faced by this paper, since the restriction is imposed on all hosts similarly. Thus, again based on the referenced work, follows the final definition. Let $n \in \mathbb{Z}^+$, $s \in \mathbb{Z}^+ \bigcup \{0\}$ and the second-order restricted set R_n^2, such that $R_n^2 = (R_0^1, R_1^1, \ldots, R_{n-1}^1)$, where $0 \le R_i^1 \le s$. The second-order restricted weak composition $C_{s,n}^{R_n^2}$ is