Future Generation Computer Systems 54 (2016) 82–94

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Power-efficient assignment of virtual machines to physical machines☆

Jordi Arjona Aroca [b,*], Antonio Fernández Anta [a], Miguel A. Mosteiro [c], Christopher Thraves [d], Lin Wang [e,f]

[a] Institute IMDEA Networks, Madrid, Spain
[b] Universidad Carlos III de Madrid, Madrid, Spain
[c] Department of Computer Science, Kean University, Union, NJ, USA
[d] CNRS-LAAS and Univ. Toulouse - LAAS, Tolouse, France
[e] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[f] University of Chinese Academy of Sciences, Beijing, China

HIGHLIGHTS

- Formal definition of 4 versions of the Virtual Machine Assignment problem (VMA).
- Study of hardness of the offline version.
- Study of competitiveness of the online version.
- Proposal of algorithms for the online version.

ARTICLE INFO

Article history:
Received 1 October 2014
Received in revised form
14 January 2015
Accepted 15 January 2015
Available online 30 January 2015

Keywords:
Cloud computing
Generalized assignment
Scheduling
Load balancing

ABSTRACT

Motivated by current trends in cloud computing, we study a version of the generalized assignment problem where a set of virtual processors has to be implemented by a set of *identical* processors. For literature consistency, we say that a set of virtual machines (VMs) is assigned to a set of physical machines (PMs). The optimization criterion is to minimize the power consumed by all the PMs. We term the problem Virtual Machine Assignment (VMA). Crucial differences with previous work include a variable number of PMs, that each VM must be assigned to exactly one PM (i.e., VMs cannot be implemented fractionally), and a minimum power consumption for each active PM. Such infrastructure may be strictly constrained in the number of PMs or in the PMs' capacity, depending on how costly (in terms of power consumption) it is to add a new PM to the system or to heavily load some of the existing PMs. Low usage or ample budget yields models where PM capacity and/or the number of PMs may be assumed unbounded for all practical purposes. We study four VMA problems depending on whether the capacity or the number of PMs is bounded or not. Specifically, we study hardness and online competitiveness for a variety of cases. To the best of our knowledge, this is the first comprehensive study of the VMA problem for this cost function.

1. Introduction

The current pace of technology developments, and the continuous change in business requirements, may rapidly yield a given proprietary computational platform obsolete, oversized, or insufficient. Thus, outsourcing has recently become a popular approach to obtain computational services without incurring in amortization costs. Furthermore, in order to attain flexibility, such service is usually virtualized, so that the user may tune the computational platform to its particular needs. Users of such service need not to be aware of the particular implementation, they only need to specify the virtual machine they want to use. This conceptual approach to outsourced computing has been termed *cloud computing*, in reference to the cloud symbol used as an abstraction of a complex infrastructure in system diagrams. Current examples of cloud computing providers include Amazon Web Services [1], Rackspace [2], and Citrix [3].

☆ A preliminary version of this work [47] was presented at the ARMS-CC 2014 workshop [48].

* Corresponding author.
E-mail addresses: jorge.arjona@imdea.org (J. Arjona Aroca), antonio.fernandez@imdea.org (A. Fernández Anta), mmosteir@kean.edu (M.A. Mosteiro), cbthraves@gsyc.es (C. Thraves), wanglin@ict.ac.cn (L. Wang).

http://dx.doi.org/10.1016/j.future.2015.01.006
0167-739X/

Depending on what the specific service provided is, the cloud computing model comes in different flavors, such as *infrastructure as a service*, *platform as a service*, *storage as a service*, etc. In each of these models, the user may choose specific parameters of the computational resources provided. For instance, processing power, memory size, communication bandwidth, etc. Thus, in a cloud-computing service platform, various ***virtual machines (VM)*** with user-defined specifications must be implemented by, or ***assigned to***,[1] various ***physical machines (PM)***.[2] Furthermore, such a platform must be scalable, allowing to add more PMs, should the business growth require such expansion. In this work, we call this problem the ***Virtual Machine Assignment (VMA)*** problem.

The optimization criteria for VMA depends on what the particular objective function sought is. From the previous discussion, it can be seen that, underlying VMA, there is some form of bin-packing problem. However, in VMA the number of PMs (i.e., bins for bin packing) may be increased if needed. Since CPU is generally the dominant power consumer in a server [4], VMA is usually carried out according to CPU workloads. With only the static power consumption of servers considered, previous work related to VMA has focused on minimizing the number of active PMs (cf. [5] and the references therein) in order to minimize the total static energy consumption. This is commonly known as VM consolidation [6,7]. However, despite the static power, the dynamic power consumption of a server, which has been shown to be superlinear on the ***load*** of a given computational resource [8,9], is also significant and cannot be ignored. Since the definition of load is not precise, we borrow the definition in [4] and define the load of a server as the amount of active cycles per second a task requires, an absolute metric independent of the operating frequency or the number of cores of a PM. The superlinearity property of the dynamic power consumption is also confirmed by the results in [4]. As a result, when taking into account both parts of power consumption, the use of extra PMs may be more efficient energy-wise than a minimum number of heavily-loaded PMs. This inconsistency with the literature in VM consolidation has been supported by the results presented in [4] and, hence, we claim that the way consolidation has been traditionally performed has to be reconsidered. In this work, we combine both power-consumption factors and explore the most energy-efficient way for VMA. That is, for some parameters $\alpha > 1$ and $b > 0$, we seek to minimize the sum of the α powers of the PMs loads *plus* the fixed cost b of using each PM.

Physical resources are physically constrained. A PMs infrastructure may be strictly constrained in the number of PMs or in the PMs CPU capacity. However, if usage patterns indicate that the PMs will always be loaded well below their capacity, it may be assumed that the capacity is unlimited. Likewise, if the power budget is very big, the number of PMs may be assumed unconstrained for all practical purposes. These cases yield 4 VMA subproblems, depending on whether the capacity and the number of PMs is limited or not. We introduce these parameters denoting the problem as ($\boldsymbol{C}$, $\boldsymbol{m}$)-**VMA**, where C is the PM CPU capacity, m is the maximum number of PMs, and each of these parameters is replaced by a dot if unbounded.

In this work, we study the hardness and online competitiveness of the VMA problem. Specifically, we show that VMA is NP-hard *in the strong sense* (in particular, we observe that (C, m)-VMA is strongly NP-complete). Thus, VMA problems do not have a fully polynomial time approximation scheme (FPTAS). Nevertheless, using previous results derived for more general objective functions, we notice that ($\cdot$, m)- and ($\cdot$, $\cdot$)-VMA have a polynomial time approximation scheme (PTAS). We also show various lower and upper bounds on the offline approximation and the online competitiveness of VMA. Rather than attempting to obtain tight bounds for particular instances of the parameters of the problem (C, m, α, b) we focus on obtaining *general bounds*, whose parameters can be instantiated for the specific application. The bounds obtained show interesting trade-offs between the PM capacity and the fixed cost of adding a new PM to the system. To the best of our knowledge, this is the first VMA study that is focused on power consumption.

Roadmap. The paper is organized as follows. In what remains of this section, we define formally the ($\cdot$, $\cdot$)-VMA problem, we overview the related work, and we describe our results in detail. Section 2 includes some preliminary results that will be used throughout the paper. The offline and online analyses are included in Sections 3 and 4 respectively. Section 5 discusses some practical issues and provides some useful insights regarding real implementation.

1.1. Problem definition

We describe the ($\cdot$, $\cdot$)-VMA problem now. Given a set $S = \{s_1, \ldots, s_m\}$ of $m > 1$ identical physical machines (PMs) of capacity C; rational numbers μ, α and b, where $\mu > 0$, $\alpha > 1$ and $b > 0$; a set $D = \{d_1, \ldots, d_n\}$ of n virtual machines and a function $\ell : D \to \mathbb{R}$ that gives the CPU load each virtual machine incurs,[3] we aim to obtain a partition $\pi = \{A_1, \ldots, A_m\}$ of D, such that $\ell(A_i) \leq C$, for all i. Our objective will be then minimizing the power consumption given by the function

$$P(\pi) = \sum_{i \in [1,m]: A_i \neq \emptyset} \left(\mu \Big(\sum_{d_j \in A_i} \ell(d_j) \Big)^{\alpha} + b \right). \tag{1}$$

Let us define the function $f(\cdot)$, such that $f(x) = 0$ if $x = 0$ and $f(x) = \mu x^{\alpha} + b$ otherwise. Then, the objective function is to minimize $P(\pi) = \sum_{i=1}^{m} f(\ell(A_i))$. The parameter μ is used for consistency with the literature. For clarity we will consider $\mu = 1$ in the rest of the paper. All the results presented apply for other values of μ.

We also study several special cases of the VMA problem, namely (C, m)-VMA, (C, $\cdot$)-VMA, ($\cdot$, m)-VMA and ($\cdot$, $\cdot$)-VMA. (C, m)-VMA refers to the case where both the number of available PMs and its capacity are fixed. ($\cdot$, $\cdot$)-VMA, where ($\cdot$) denotes unboundedness, refers to the case where both the number of available PMs and its capacity are unbounded (i.e., C is larger than the total load of the VMs that can ever be in the system at any time, or m is larger than the number of VMs that can ever be in the system at any time). (C, $\cdot$)-VMA and ($\cdot$, m)-VMA are the cases where the number of available PMs and their capacity is unbounded, respectively.

1.2. Related work

To the best of our knowledge, previous work on VMA has been only experimental [10–15] or has focused on different cost functions [16,17,5,18]. First, we provide an overview of previous theoretical work for related assignment problems (storage allocation, scheduling, network design, etc.). The cost functions considered in that work resemble or generalize the power cost function under consideration here. Secondly, we overview related experimental work.

Chandra and Wong [18], and Cody and Coffman [16] study a problem for storage allocation that is a variant of ($\cdot$, m)-VMA with $b = 0$ and $\alpha = 2$. Hence, this problem tries to minimize the sum

[1] The cloud-computing literature use instead the term *placement*. We choose here the term assignment for consistency with the literature on general assignment problems.

[2] We choose the notation VM and PM for simplicity and consistency, but notice that our study applies to any computational resource assignment problem, as long as the minimization function is the one modeled here.

[3] For convenience, we overload the function $\ell(\cdot)$ to be applied over sets of virtual machines, so that for any set $A \subseteq D$, $\ell(A) = \sum_{d_j \in A} \ell(d_j)$.