Microprocessors and Microsystems 000 (2016) 1–16

Contents lists available at ScienceDirect

Microprocessors and Microsystems

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

Joint affinity aware grouping and virtual machine placement

Jianhai Chen[a], Qinming He[a], Deshi Ye[a,*], Wenzhi Chen[a], Yang Xiang[b], Kevin Chiew[c], Liangwei Zhu[a]

[a] College of Computer Science, Zhejiang University, Hangzhou, China
[b] School of Information Technology, Deakin University, Burwood, Vic. 3125, Australia
[c] Handal Indah Pte. Ltd., Singapore

A R T I C L E I N F O

Article history:
Received 2 February 2016
Revised 25 November 2016
Accepted 9 December 2016
Available online xxx

Keywords:
Virtualization
Affinity grouping
Resource allocation
VM colocation
Affinity scheduling
Bin packing

A B S T R A C T

The **N**on-**A**ffinity **A**ware **G**rouping based resource **A**llocation (NAGA) method toward the ***G**eneral VM**P**lacement (GP) problem* enables (1) some VMs to be co-located onto the same PM while the VMs are required to be placed onto distinct PMs; and (2) some VMs to be dispersedly placed onto distinct PMs while the VMs are required to be co-located onto the same PM, leading to a serious performance degradation of application running over multiple VMs in cloud computing. In this work we study an ***A**ffinity **A**ware VM **P**lacement (AAP)* problem and propose a ***J**oint **A**ffinity **A**ware**G**rouping and **B**in **P**acking (JAGBP)* method to remedy the deficiency of the NAGA method. We firstly introduce affinity of VMs to identify affinity relationships to VMs which are required to be placed with a special VM placement pattern, such as colocation or disperse placement, and formulate the AAP problem. Then, we propose an affinity aware resource scheduling framework, and provide methods to obtain and identify the affinity relationships between VMs, and the JAGBP method. Lastly, we present holistic evaluation experiments to validate the feasibility and evaluate the performance of the proposed methods. The results demonstrate the significance of introduced affinity and the effectiveness of JAGBP method.

1. Introduction

Cloud computing has been a paradigm for delivering computing services to users with an abstraction of scalable, unlimited computing resources on a pay-as-you-go basis and it runs applications to provide services for users on remote datacenters over the Internet [9]. Nowadays many large companies, such as Amazon, Google, Microsoft and Alibaba, have built public clouds successfully. More and more enterprises set up private clouds managed by frameworks such as VMware vCloud [1], OpenStack [5], and CloudStack [18], paving a new commercial business model with migration of business applications to clouds. Due to a recent survey of IT decision makers of large companies, 68% of the respondents estimate that more than 50% of their companies' IT services will be transmitted to cloud platforms by the end of 2014 [26].

The success of cloud computing partly ascribes to virtualization. Virtualization has become a crucial technology of cloud computing and built the essential infrastructure for clouds. It enables server consolidation and live migration, and brings the dynamical resource scheduling and allocation into reality for clouds. Various applications, such as High Performance Computing (HPC) applications [21], multi-tiers web applications [17], parallel applications [14], and big data processing applications [2], are encapsulated into multiple virtual machines (VMs) which are dynamically allocated to a large pool of physical machines (PMs) [8,32,44].

With the growing number of users using clouds, the VM placement resource scheduling and allocation has become an increasingly important problem of current datacenter [20]. Many cloud datacenters currently are still at a very low resource utilization and need efficient resource allocation methods [6]. The goal of resource allocation is commonly to maximize datacenter revenues or the ratio between performance and cost. It involves performance efficiency as well as energy efficiency. The performance efficiency requires the allocation to minimize application performance cost using virtualization. The energy efficiency requires the allocation to maximize resource utilization, namely, minimize the number of PMs.

Besides, as virtualization brings itself performance cost, some VMs running applications are required to be placed with a specific placement pattern, i.e., VMs colocation or disperse placement [15,30]. For examples, the running of many communication-/data- intensive applications inside multiple VMs generates net-

* Corresponding author.
E-mail addresses: chenjh919@zju.edu.cn (J. Chen), hqm@zju.edu.cn (Q. He), yedeshi@zju.edu.cn (D. Ye), chenwz@zju.edu.cn (W. Chen), yang@deakin.edu.au (Y. Xiang), kchiew@handalindah.com.my (K. Chiew), zhulw@zju.edu.cn (L. Zhu).

http://dx.doi.org/10.1016/j.micpro.2016.12.006
0141-9331/© 2016 Published by Elsevier B.V.

work communication traffics or data traffics amongst VMs [25]. Colocating VMs with traffics can reduce network communication overhead and improve application performance. While co-locating the VMs onto the same PM can cause competition of shared computing resources, such as CPU, memory, disk and network I/O bandwidth, etc. The special demand of VM placement pattern brings a dependency or relationship for VMs running applications.

Nevertheless, considering the dependency between VMs, the VM placement resource scheduling and allocation bring forward two major challenges:

(1) Obtaining the dependency of VMs with an effective placement pattern for running applications efficiently is not a trivial task. It requires a detailed analysis of application program behaviours or features and a holistic performance evaluation for running application under all kinds of distinct VM placement schemes.
(2) Solving the problem of VM placement resource scheduling and allocation is always an optimization problem, such as the bin packing problem. Due to bin packing is an NP-hard problem, making decision of an optimal VM placement resource allocation solution for both minimizing the number of PM to guarantee energy efficiency and guaranteeing the dependency between VMs is a big challenge.

There have been some research works addressing the challenges and proposed some methods on resource allocation [10,34,39–42]. However, current methods rarely consider the dependency of VMs and lack a general approach to generalize the dependency of VMs for VM placement. Further, there still are no general efficient methods to obtain and identify relationships between VMs in VM placement. Although some methods solve the general VM placement resource allocation as a bin packing problem, without consideration of the dependency or relationship between VMs, the bin packing methods will enable the VMs which are required to be colocation placed onto the same PM but not, and the VMs which are required to be dispersedly placed onto distinct PMs but not, causing a certain level of application performance degradation [46].

In this paper, we target to remedy the deficiency of the general resource allocation methods like bin packing and focus on both minimizing the application performance cost and maximizing the resource utilization. At first, by conducting experiments in a testbed with several typical cloud applications running between multiple VMs, we do performance evaluation by running applications under two different VM placement patterns, namely, VMs colocation placement and dispersedly placement with all kinds of combination of VMs and PMs. Motivated by the observation results, we introduce affinity to denote the dependency between VMs and define affinity relationships between VMs for VM placement and state an affinity aware VM placement (AAP) problem. Then, we propose a *Joint Affinity Aware Grouping and Bin Packing* (JAGBP) method to solve the AAP problem, including an affinity grouping algorithm and several heuristic bin packing algorithms. At last, we present three experiments to validate efficiency of the proposed methods, including (1) experiments on obtaining and identifying the affinity relationships between VMs, (2) a simulation experiment on verifying the run efficiency of algorithms, and (3) a real cloud environment experiment to illustrate the effectiveness of the JAGBP method through constructing seven virtual clusters (VCs) configured with 59 VMs for running typical cloud applications, including the HPCC and RUBiS benchmarks. By comparing to the Non-affinity aware grouping allocation (NAGA) method, the JAGBP method significantly improve the application performance better than the NAGA method.

In brief, our contributions are as follows.

(1) We model affinity between VMs for VM placement. In the model, we give the methods to find and identify affinity relationships between VMs, involving the performance measurement, analysis and evaluation of typical cloud virtualization applications.
(2) We present an affinity aware VM placement (AAP) problem and the JAGBP method. To the best of our knowledge, we are the first to present AAP problem and propose JAGBP method.
(3) We provide an affinity aware VM placement framework for cloud computing system resource scheduling. It integrates affinity obtaining methods and resource scheduling algorithms which can be efficiently used to advance the practical cloud computing platforms.
(4) We conduct overall experiments to demonstrate the effectiveness of our proposed methods.

The remaining sections are organized as follows. We present the related work in Section 2, the background and motivation of our work in Section 3. Next, we model the affinity and affinity relationship, and state the affinity aware placement problem in Section 4 and we propose solutions to solve the problem in Section 5, followed by experiments in Section 6. Finally we give conclusion in Section 7.

2. Related work

This paper presents an approach for VM placement resource allocation in cloud computing systems, considering both optimization of application performance and resource utilization to provide highly performance cost ratio for cloud. Here, we discuss related work in the literature related to similar issues.

2.1. Affinity and virtualization performance studies

Many research on affinity and virtualization performance has been conducted in cloud computing.

Chen and Li et al. proposed affinity as a property of VMs to identify the relation between a virtual CPU and CPU core in VMM or Hypervisor to implement a new schedule strategy to improve the efficiency of virtualized resource scheduling under a single virtualized system in cloud computing [4]. Sonnek et al. presented an affinity-aware VM migration technique to minimize the communication overhead on a virtualized platform [33]. The affinity is identified as a policy or a technique of VM migration for a dynamic resource allocation. Yan, Cairong et al. discussed an affinity aware virtual cluster optimization method for Mapreduce applications placement [43]. The affinity is defined as relationships between virtual clusters and obtained by measuring the latency distance between network virtual machines. The proposed affinity in this paper denotes relationships between VMs associated with special VM placement patterns, colocation placement and no-colocation placement.

Sudevalayam et al. put forward an affinity aware modeling of CPU usage method to evaluate performance of virtualized applications hosted onto two VMs with colocation placement or disperse placement [35]. They focus on CPU usage-based application performance evaluation under VM colocation or disperse location scenarios but not resource allocation. Two years later, they described an affinity aware modeling of CPU usage with communicating virtual machines and develop pair-wise affinity-aware models to predict expected CPU resource requirements [36]. Their proposed performance evaluation methods paved a good way to obtain affinity of VMs for our work. Our work presents a general affinity of VMs and affinity relation model for VM placement resource allocation. Meng et al. introduced a traffic-aware VM placement to improve the network scalability [25]. The network traffic generates communications. The authors did not address affinity for VM placement but the traffic property defines a kind of affinity for our work. Re-