www.ijiarec.com

ISSN:2348-2079

Volume-7 Issue-2

International Journal of Intellectual Advancements and Research in Engineering Computations

An Efficient Resource Allocation for Dynamic Workflow in IaaS Cloud

S.Monisha[1], S.Mounika[2], K.Sruthy Sudhakar[2], Mr.S.Prabhu[2]

[1]UG Students, Department of Computer Science and Engineering ,Nandha Engineering College, Erode, Tamilnadu, India.

[2]Assistant Professor, Department of Computer Science and Engineering ,Nandha Engineering College, Erode, Tamilnadu, India.

ABSTRACT

Infrastructure-as-a-service (IaaS) cloud technology has attracted much attention from users who have demands on large amounts of computing resources. Current IaaS clouds provision resources in terms of virtual machines (VMs) with homogeneous resource configurations where different types of resources in VMs have similar share of the capacity in a physical machine (PM). However, most user jobs demand different amounts for different resources. For instance, high- performance-computing jobs require more CPU cores while big data processing applications require more memory. The existing homogeneous resource allocation mechanisms cause resource starvation where dominant resources are starved while non-dominant resources are wasted. To overcome this issue, we propose a heterogeneous resource allocation approach, called skew ness-avoidance multi-resource allocation (SAMR), to allocate resource according to diversified requirements on different types of resources. Our solution includes a VM allocation algorithm to ensure heterogeneous workloads are allocated appropriately to avoid skewed resource utilization in PMs, and a model- based approach to estimate the appropriate number of active PMs to operate SAMR. We show relatively low complexity for our model based approach for practical operation and accurate estimation. Extensive simulation results show the effectiveness of SAMR and the performance advantages over its counterparts.

Keywords: Cloud Computing, Scheduling, Optimization, Workflows, Resource capacity, Resource provisioning, Deadline, Virtualization, Virtual Machine, Auto-scaling, Consolidation, Mutation.

INTRODUCTION

Cloud computing is a technology that make use of the Internet to obtain software or other IT services on demand. This on-demand computing power and storage capacity makes cloud as an efficient computing platform. Cloud computing provides the shared resources needed by the users at any given time by pay as they go pricing scheme and keeping cost to the user down. Cloud computing is a technology and business model as well as where the providers are providing the software, hardware, platform, or storage providers, deliver their offerings over the Internet. The data application of providers is maintained by the central remote servers by using internet. The applications are used without installation and the personal files which can be located at different location form user can be access through internet. Cloud computing is not a new technology but it leverages the advantages of existing and increases the efficiency. When compared to grid computing which coordinates the networked resources as the distributed computing paradigm, the cloud computing satisfies this distributed resources objective like computing power, software, storage services, and platforms by virtualization technologies which helps user to complete the jobs in less time and cost. When compared to utility computing which provides on demand resources and charge based on the usage, the cloud computing

Author for correspondence:
Department of Computer Science and Engineering, Nandha Engineering College

perceived this utility based scheme by increasing the resource utilization and decrease the operating cost. Virtualization is the key feature that acts as the foundation for providing the resources to users on demand. The commonly used virtualized server called virtual machine (VM) helps for assigning and reassigning the resources on demand. These services can be provides by cloud from anywhere and at any time through internet. The cloud computing technologies and virtualization growth requires the virtual machines for number of jobs. Since cloud computing becomes the emerging and attractive technology, many user deploy their application in the cloud or extend their home clusters during high demand results in workflow management system. The workflow management system should balance both performance and cost. Different types of scheduling optimization approaches have been developed to reduce the cost and increase the performance. In this paper the cost based task scheduling and dynamically optimized resource allocation strategies has been surveyed. Public clouds have attracted much attention from both industry and academia recently. Users are able to benefit from the clouds by highly elastic, scalable and economical resource utilizations. By using public clouds, users no longer need to purchase and maintain sophisticated hardware for the resource usage in their peak load. In recent years, many efforts have been devoted to the problem of resource management in IaaS public clouds such as Amazon EC2 and Rack space cloud. All these works have shown their strength in some specific aspects in resource scheduling and provisioning. However, existing works are all on the premise that cloud providers allocate virtual machines (VMs) with homogeneous resource configurations. Specifically, homogeneous resource allocation offers resources in terms of VMs where all the resource types have the same share of the physical machine (PM) capacity. Both dominant resource and non-dominant resource are allocated with the same share in such manner even if the demands for different resources from a user are different.

RELATED WORKS

In this section, we organize the literature review for resource management in clouds in two main categories: homogeneous resource allocation and heterogeneous resource allocation. S. Genaud and J. Gossa, [1] presents how billing models can be exploited by provisioning strategies to find a trade-off between fast/expensive computations and slow/cheap ones for independent sequential jobs. We simulate these strategies on real grid workloads in two cases. First, we use the workloads as a whole, which is representative of a large community of users sharing some common resources. Second, we use the workloads extracted for each individual user. The need of rewriting source codes, previous knowledge of the application and/or stop-reconfigure-and-go approaches to address cloud elasticity has been proposed in [2]. Besides, there are problems related to how profit this new feature in the HPC scope since in MPI2.0 applications the programmers need to handle communicators by themselves and a sudden consolidation of a VM, together with a process, can compromise the entire execution. An elastic environment that extends a local cluster resource with IaaS resources has been proposed in [3]. They present resource provisioning policies to dynamically match resource supply with demand. Our policies balance the requirements of users and administrators, such as minimizing the monetary cost of the IaaS deployment and reducing job queued time. We develop a discrete event simulator, the elastic cloud simulator (ECS), to evaluate our policies using scientific workloads. An enhanced scientific public cloud model (*ESP*) that encourages small or medium scale research organizations rent elastic resources from a public cloud provider; second, on a basis of the ESP model, we design and implement the *Dawning Cloud* system that can consolidate heterogeneous scientific workloads on a Cloud site; third, we propose an innovative emulation methodology and perform a comprehensive evaluation has been proposed in [6]. [5] Aims to offer this analysis from a client-side scheduling perspective in which emphasis is not put on physical resource selection but on task to virtual machine mappings and virtual machine allocation. It provides one tax onomy for the current state of the art and one unified model concerning the various metrics and goals used throughout literature. [6] Addresses a new and

important problem concerning the efficient management of such ensembles under budget and deadline constraints on Infrastructure- as-aService (IaaS) clouds. We discuss, develop, and assess algorithms based on static and dynamic strategies for both task scheduling and resource provisioning. We perform the evaluation via simulation using a set of scientific workflow ensembles with a broad range of budget and deadline parameters, taking into account uncertainties in task runtime estimations, provisioning delays, and failures. We find that the key factor determining the performance of an algorithm is its ability to decide which workflows in an ensemble to admit or reject for execution. [7] Propose an algorithm that can enlarge the chance of selecting the best policy in limited time, possibly online. Through trace-based simulation, we evaluate various aspects of our portfolio scheduler, and find performance improvements from 7% to 100% in comparison with the best constituent policies and high improvement for bursty workloads.[8] propose a cost effective and dynamic VM allocation model based on Nash bargaining solution. With various simulations it is shown that the proposed mechanism can reduce the overall cost of running servers while at the same time guarantee QoS demand and maximize resource utilization in various dimensions of server resources.[9] argues a more flexible approach that IaaS providers should offer virtual machines with flexible combinations on multiple resource types. We further formulate the problem of multiple resource virtual machine allocations for IaaS clouds, and develop analytical models to predict the suitable number of PMs While satisfying a predefined qualityof-service requirement. Experiments show that the proposed approach can significantly increase the resource utilization, with a reduction on the number of active PMs by 27% on average.[10]The workload is also highly dynamic, varying over time and most workload features, and is driven by many short jobs that demand quick scheduling decisions. While few simplifying assumptions apply, we find that many longer-running jobs have relatively stable resource utilizations, which can help adaptive resource schedulers. [11] Our solution includes a VM allocation algorithm to ensure heterogeneous workloads are allocated appropriately to avoid skewed resource utilization in PMs, and a model-based approach to estimate the appropriate number of active PMs to operate SAMR. We show relatively low complexity for our model-based approach for practical operation and accurate estimation. Extensive simulation results show the effectiveness of SAMR and the performance advantages over its counterparts.[12] Elastic resource allocation in cloud computing paradigm is managing the resource elastically from anywhere and in anytime automatically. Elasticity is the fundamental characteristic by combining on-demand self service and resource pooling techniques in cloud, distributed, grid and utility computing circumstance. In addition to, this paper discourse open challenges and future directions connected with role of elasticity mechanism in cloud computing [13] we aim to minimize such performance degradation. Moreover, 2 overloaded VMs tend to steal resources (e.g. CPU) from neighbouring VMs, so our work maximizes VMs real CPU utilization. Based on these, we provide an experimental analysis to compare our solution with traditional schedulers used in Open Stack by exploring the behaviour of different NoSQL (MongoDB, Apache Cassandra and Elastic search). The results show that our solution refines traditional instant-based physical machine selection as it learns the system behaviour as well as it adapts over time. The analysis is prosperous as for the selected setting we approximately minimize performance degradation by 19% and we maximize CPU real time by 2% when using real world workloads.[14] proposed a hybrid provisioning for both HPC and cloud workloads to cover their features in resource allocation (HPC jobs are all queued by the scheduling system, but jobs in public clouds use all-or-nothing policy). However, these approaches only consider CPU as the dominant resource in single-dimensional resource allocation. To handle the provisioning problem for heterogeneous workloads, this paper proposes a model- based provisioning method that provisions minimum amount of resources while satisfying the allocation delay constraint. [15] both designed similar elastic PM provisioning strategy based on predicted workloads. They adjust the number of PMs by consolidating VMs in over-provisioned cases and powering on extra PMs in under- provisioned cases. Such heuristic adjusting is

simple to implement, but the prediction accuracy is low.

SYSTEM OVERVIEW

In this section, we introduce the application scenario of our research problem and provide a system overview on our proposed solution for heterogeneous resource allocation. Table 1 lists the key notations used throughout this paper. Similar to other works that optimize the resource usages in the clouds we use the number of active PMs as the main metric to measure the degree of energy consumption in clouds. Reducing the number of active PMs in data centre to serve the same amount of workloads with similar performance to users is of great attraction for cloud operators.

TABLE 1

Notations used in algorithms and models

K	**Number of resource types**
N_{total}	Total number of PMs in the considered data center
$\vec{R}$	$r_i$is the capacity of type-iresource in a PM, i= [1,2,...,K]
M~	m_i, $m_i < r_i$is the maximum resource for type-iresource in a VM, i= [1,2,...,K]
X	Total number of VM types
V~ x	The resource configuration of type-x VM, v^x_i($v^x_i \leq m_i$) represents the amount of type-iresource, x = [1,2,...,X] and i= [1,2,...,K]
C~	c_i is the total consumed type-iresource in a PM, $c_i \leq r_i$, i= [1,2,...,K]
U~	$u_i$is the utilization of type-iresource in a PM, $u_i \in$ [0,1], i= [1,2,...,K]
Λx	Arrival rate of type-x requests, x = [1,2,...,X]
μx	Service rate of type-x requests, x = [1,2,...,X]
D	Predefined VM allocation delay threshold
D	Actual average VM allocation delay in a time slot
N	Provisioned number of active PMs (predicted by the model)
S~	$s_n$is the skewness factor for n^{th} active PM, n = [1,2,...,N]

Table1. Notations used in algorithms and models

We consider the scenario where cloud users rent VMs from IaaS public clouds to run their applications in a pay-as-you-go manner. Cloud providers charge users according to the resource amounts and running time of VMs. Fig. 2 shows the system model of our proposed heterogeneous resource allocation approach SAMR. Generally, we assume that a cloud data centre with N_{total}PMs offers K different resource types (e.g., CPU, RAM, Disk, ...). The cloud system offers X different VM types, each of which is with a resource combination $V^{\sim x}$= {v^x_i|i= 1,2,...,K}(x = 1,2,...,X) where v^x_idenotes the resource capacity of i^{th} resource type in x^{th} VM type. Cloud users submit their VM requests (also denoted as workloads in this paper) to the cloud data centre according to their heterogeneous resource demands and choose the VM types that are most appropriate in terms of satisfying the user demands while minimizing the resource wastage. We refer a request for x^{th} type of

VM as a type-x request in workloads. All VM requests are maintained by a scheduling queue. For each request from users, resource (or VM) scheduler allocates the resources for requested VM in N current active PMs if the resource slot of the VM is available. Otherwise, the request will be delayed waiting for more PMs to power up and join the service. According to the arrival rates and service rates of requests, SAMR conducts resource prediction based on a Markov Chain model periodically in every time slot with a duration of t to satisfy the user experience in terms of VM allocation delay. By such manner, we focus on solving the problem in a small time period to increase the prediction accuracy. After the online prediction of required resources, the cloud system provisions corresponding number of active PMs N in the coming time slot. In VM scheduling phase during each time slot with the length t, cloud providers allocate resources and host each VM into PMs using SAMR allocation algorithm.

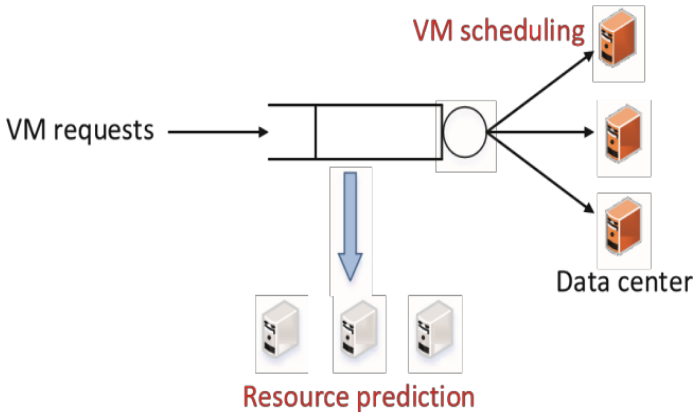

Fig. 2. System architecture of SAMR.

In cloud service, one of the most significant impacts on user experience is the service delay caused by schedulers. Here we consider the resource (or VM) allocation delay as the main metric for service level agreements (SLA) between users and cloud providers. Specifically, SAMR uses a VM allocation delay threshold D to be the maximum SLA value that cloud providers should comply with. Thus, there is a trade off between cost and SLA for cloud providers. To cope with the large amount of random request arrivals from users, it is important to provision enough active PMs. However, maintaining too many active PMs may cope well even under peak load but wastes energy unnecessary. Maintaining too few PMs may cause significant degradation in user experience due to lacks of active PMs and the need to wait for powering up more PMs. It is challenging to find the adequate number of active PMs. In our work, during the resource prediction phase, SAMR uses a Markov Chain model to find the adequate number of active PMs that satisfies the SLA value. Precisely, the model determines the number of active PMs, N, such that the average VM allocation delay d is smaller than the agreed threshold D. We use the Markov Chain model to determine the adequate number of active PMs for operation. The model assumes heterogeneous workloads and balanced utilization of all types of resources within a PM. To realize the balanced utilization, we define a multi-resource skeins as the metric to measure the degree of unbalancing among multiple resource types as well as multiple PMs. The SAMR scheduling aims to minimize the skew ness in data centre in order to avoid the resource starvation.

EXISTING SCHEDULING ALGORITHM ON

Performance And Cost Optimization

Adaptive Dual Objective Scheduling Algorithm(Ados)

The basic objective of this effort is to minimize the completion time of the task. However

it may causes the redundant usage of resources, which results in the novel scheduling algorithm f the solution. This algorithm gives better result in completion time and also in resource usages. Due to dynamic changes in the resource, the performance fluctuation may occur which can be managed by rescheduling. The operation starts with the initial random schedule by branch and bound technique and a genetic operator which consists of point and swap mutation. The make span and resource usage has given equal importance including the rescheduling which increases the feasibility of the ADOS algorithm [1]. The scheduling problem is addressed as scheduling of mutually dependent jobs with workflow applications onto a set of heterogeneous hosts. This scheduling process mainly focuses on finding appropriate job that matches to reduces the make span or schedule length. If the matches are better than the original then it is selected and produced good quality schedule on small resource usage. The loosened schedule selection scheme has been included to produce an effective solution which reduces the makes pan significantly. The algorithm randomly selects the mutation method to essentially select different schedules on probability 0.5. The mutation occurs only during the rescheduling process. The mutated schedules are used as current schedule which continues best in the next iteration. The dual objective approach has been executed by achieving static heurist

Partition balanced time scheduling algorithm(PBTS)

The important challenge in integrating workflow with resource provisioning is to find the appropriate amount of resources needed to execute the task and reduce the cost from perspective of users and increase the resource utilization from perspective of resource provider. The algorithm evaluated the minimum number of computing hosts for executing the workflow within the user specified finish time. The resource capacity is the amount of resource request by workflow to provisioning system. When resource capacity is high the make span reduces but results in low resource utilization and high cost. When resource capacity is low, the execution time increases. The first heuristic algorithm called Balanced Time Scheduling (BTS) which evaluate minimum resources needed to execute workflow within specified completion time. Since BTS is static for complete workflow execution, then the extended polynomial algorithm called Partition Balanced Time Scheduling (PBTS) [2] which estimates the best number of resources part time charge unit. This process reduces the gross cost during the application lifetime. PBTS is the scalable approach which can handle workflow with date-parallel tasks and MPI like: Parallel tasks whose sub-tasks are executed concurrently on various resources .PBTS utilizes the elasticity of resources and gives the execution plan with lower cost. PBTS is dynamic which adjust the resource capacity to lower cost and completed within specified time.

Auto scaling Mechanism

The auto scaling mechanism [3] faces many challenges regarding performance and cost. Additional virtual machine needs can be acquired any time but it take time to use those VMs. As a result the cloud providers and Third Party Auditor provides schedule-based and rule- based mechanisms. These mechanisms auto starts with auto scaling which scales up and scales down the VMs. These are simple but not conveying enough to address requirements of users. The approach has the computing elements as VM and their cost. The performance requirements are workflow and deadline. The aim of this approach is to complete the task with minimum cost.

Transformation Based Ptimization Framework (DAG)

The cloud computing is the pay-as –you-go model which gives more important to metrics like cost and performance. This becomes the great challenge in the workflow and the performance system. Due to interconnected factors in the workflow, the performance and cost optimization became the important metrics. Requirements are differ based on the users which focus only on cost and compromise with performance. Some of them force on performance and compromise with budget. All these techniques lacks on optimization techniques on cost and performance. Then the transformation based optimization framework[5]

has been processed to optimize the performance and cost. The directed acyclic graph(DAG) model has been used to denote the workflow.

Workflow

Workflow structures are generally represented as DAG is G(V,E). V denotes the set of vertices is the tasks. E denotes the set of edges in the data dependencies between the tasks.

Initial Assignment

Initially a task assigned to the instance type per execution in each workflow. Various heuristics based methods are has been used to assign the instances to task which also forms as DAG called instance assignment graph.

Transformation operation

The transformation operations results in structural changes of the assignment of DAG. The transformation operations are classified as main schemes and auxiliary schemes. The main scheme aims to reduce the cost. The auxiliary schemes aim to change the form of workflow which is suitable for main scheme to reduce cost. The six basic workflow transformation operations are Merge, Demote, Split, Promote, Move and co-scheduling. The merge and demote operation comes under main scheme. The Split, Promote, Move and co-scheduling comes under the auxiliary scheme.

Merge operation

The merge operation performs when two vertices are assigned to the instances of same type. The vertices are assigned to one after another. The instance node of the instance DAG are combined to form the super node and maintain the hierarchical relationship and structural dependencies among the nodes in DAG.

Demote operation

The demote operation performs the execution of single vertex by assigning it to the cheaper instance which causes the longer execution time. The dependencies of the demote vertex also delayed by the demoted vertex also delayed by the demoted vertex.

Move operation

The moving operation is used for moving one task after the end of another task to reduce the task. The dependencies of the moved vertex were also delayed by the moved vertex.

Split operation

The split operation is performed when more urgent task need to run on the same type instance by pausing the current task for 4 a particular time. The suspended technique can be resumed by the checkpoint technique after the completion of the urgent task.

Promote operation

The promote operation is performed during the execution of the task to a better or costlier instance for decreasing the execution time. The promote operation are mainly performed to satisfy the deadlines. The promote operation continues with the merge operation to utilize the instances.

Co-scheduling operation

The co-scheduling operation is performed when multiple tasks running at the same time. The multiple tasks which have similar start time and end time with similar leftover time for deadline can be run at the same instance type.

Optimization Sequence

DAG Graph

This DAG Graph helps to find the sequence of operation to be performed for cost and performance optimization. The Graph has three designs. First the Graph runs periodically. Second the main schemes and auxiliary schemes are performed alternatively and cost model avoids the unwanted transformation operations. Third the rule is defined to achieve the performance and cost optimization goal.

The Optimization process of DAG for workflows

All workflows are allowed in the form of queue. Each workflow is initially assigned with instance type. The selected main scheme and auxiliary scheme operations are performed for largest cost reduction without affecting the deadline of the workflow. If the time constraints

are not violated then the cost has been estimated by cost model. If these transformation operations has no cost reduction then the initial assignment are executed to optimize the workflow.

Max-Min Task-Scheduling Algorithm

The main aim of task scheduling algorithm mainly in cloud computing is the allocation of task to the particular node. The allocation of task to the appropriate node is the challengeable task which can be improved by Max-Min Task Scheduling algorithm [6]. This scheduling algorithm maintains the task status table and virtual machine status table which help to identify the workload of virtual machines. This table also helps to find the task completion time.

When the tasks enter the scheduler, the scheduler finds the longest execution time tasks in the task status table then select the virtual machine in the virtual machine status table and finds the shortest completion time. This result in the allocation of tasks to the total number of tasks and their execution time in the virtual machine are updated in the virtual machine status table. This process is repeated until every task is allocated to the concerned virtual machine.

Cost with finish time-based algorithm(CWFT)

The cloud computing scheduling framework is mainly built on the basis of computing resources on cloud and computing components of our local systems. The Cost with Finish Time- based scheduling algorithm[7] focuses on increasing the performance of process and to reduce the monetary cost for resource provided by the cloud. Usually in distributed environments, the optimization techniques focuses on makespan, but the CwFT focuses on both makes pan and monetary cost. The algorithm consists of two phase.

Task prioritizing phase

The priority level of all tasks is assigned in this phase. The priority of the task is estimated on the basis of length of critical path which includes the computation time. Finally, the sorted list of all tasks in descending order is determined. The goal of this list is to provide the topological order of the tasks. This list avoids the precedence constraints.

Node selection phase

In the phase the sorted list from the task prioritizing phase are scheduled to the appropriate nodes to optimize cost and performance. The appropriate nodes are selected on the basis of the Earliest Execution Starting Time (EST). Earliest Execution Finish Time (EFT) and Data Transfer Time (DTT). Based on the parameters, the best nodes are allocated to complete the tasks.

Dynamically Optimized Cost-based Task Scheduling Algorithm

The main objective of this scheduling [11] approach is to combine the solution at user level and provider level to get the optimal solution. The cost based task scheduling aims to reduce the cost to the user. The dynamically optimized resource allocation strategy aims to benefit the service provider. The user tasks are grouped to utilize the available resources before resource allocation. In this algorithm the prioritization is based on the task profit. The execution done on descending order for the task profit. The high profit tasks are executed on the low cost machines to reduce the cost of the user. This algorithm benefits both the user and provider.

EVALUATION

In this section, we evaluate the effectiveness of our proposed heterogeneous resource allocation approach with simulation experiments. First, we introduce the experimental setups including the simulator, methods for comparison and the heterogeneous workload data. Second, we validate SAMR with simulation results and then compare the results with other methods.

Simulator maintains the resource usage of PMs in the cloud and support leasing and releasing the resources for VMs requested by users. We consider offering of two resource types: CPU cores and memory. In our experiments, we set the time for powering on a PM to 30 seconds and the default average delay constraint is set to 10 seconds. The default maximum VM capacity is set to 32% of the normalized capacity of a PM. Besides, the default time slot for resource allocation is 60 minutes. To

study their impact on system performance, sensitivities of these parameters are investigated in the experiments. We study the following performance metrics in each time slot: number of PMs per time slot, mean utilization of all active PMs, multi-resource skewness factor and average VM allocation delay. The number of PMs is the main metric which can impact the other three metrics.

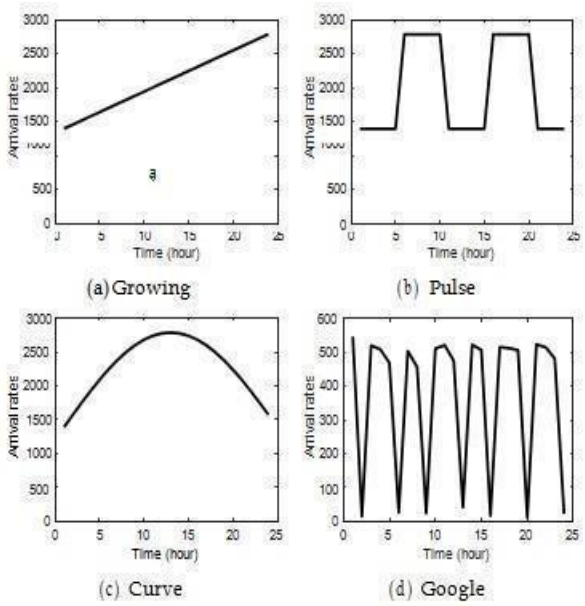

Fig (ii) Three synthetic workload patterns and one real world cloud trace from Google.

To evaluate the effectiveness of SAMR in serving highly heterogeneous cloud workloads, we simulate and compare the results of SAMR with the following methods: 1) single-dimensional (SD). SD is the basic homogeneous resource allocation approach that is used commonly in current IaaS clouds. Resource allocation in SD is according to the dominant resource, other resources have the same share of dominant resource regardless of users' demands. For scheduling policy, we simply choose first fit because different scheduling policies in SD have similar performance impact on resource usage. In first fit, the provisioned PMs are collected to form a list of active PMs and the order of PMs in the list is not critical. For each request, the scheduler searches the list for available resources for the allocation. If the allocation is successful, the requested type of VM will be created. Otherwise, if there is no PM in the list that can offer adequate resources, this request will be delayed. 2) Multi-resource (MR). Different from SD, MR is a heterogeneous resource allocation method which do not consider multi-resource skew ness factor in resource allocation. MR offers flexible resource combinations among different types of resource to cover different user demands on different resource types. MR also uses first fit policy to host VMs in cloud data centre. 3) Optimal (OPT). An optimal resource allocation (OPT) is compared as the ideal provisioning method with oracle information of workloads. OPT assumes that all PMs run with utilizations of 100%. The provisioning results of OPT are calculated simply by dividing the total resource demands in each time slot by the capacity of the PMs. Thus, OPT is considered as the most extreme case that minimum number of PMs are provisioned for the workloads.

Workloads. Two kinds of workloads are utilized, synthetic workloads and real world cloud trace, in our experiments as shown in Fig. 5. In order to study the sensitivity of performance under different workload features, three synthetic workload patterns are used: growing, pulse and

curve. By default, the lowest average request arrival rates of all three synthetic workload patterns are 1400 and the highest points are 2800. We keep the total resource demands of each type of VM requests similar so that the number of VM requests with higher resource demands is smaller. The service time of the VMs in synthetic workloads are set to exponential distribution with average value of 1 hour.

EXPERIMENTAL RESULTS

Overall results. We first present the overall results of the four methods for the four workloads. The bars in the figure show the average values for different results and the vertical red lines indicate the 95% confidence intervals.

We make the following observations based on the results. Firstly, heterogeneous resource management methods (MR and SAMR) significantly reduce resources in terms of number of active PMs for the same workloads. The resource conservation achieved by MR compared with SD is around 34% for all four workloads. SAMR further reduces the required number of PMs by another 11%, or around 45% compared with SD. It shows that SAMR is able to effectively reduce the resource usage by avoiding resource starvation in cloud data centre. Besides, the number of active PMs for SAMR is quite close to the optimal solution with only 13% difference. Note that the presented number of active PMs for SAMR is the actual required number for the given workloads. Based on our experiment records, the predicted numbers of PMs from our model have no more than 5% (4.3% on average) error rates compared with the actual required numbers presented in the figure. Secondly, although the utilization of dominant resource using SD method is high, the no dominant resources are under-utilized. However, the resource utilizations in MR and SAMR policies are balanced. This is the reason that SD must provision more PMs. Thirdly, the effectiveness of resource allocation in SAMR is validated by the skew ness factor, where the average resource skew ness factors in SAMR method are less than that in MR. Finally, all three policies achieve the predefined VM allocation delay threshold SD holds slight higher average delays than SAMR and MR, which is due to the fact that SD always reacts slowly to the workload dynamicity and cause more under-provisioned cases to make the delay longer.

Impacts by the amount of workloads shows the detailed results of all methods for different metrics under four workloads. We highlight and analyse the following phenolmenon's in the results. Firstly, heterogeneous resource allocation methods significantly reduce the required number of PMs in each time slot for 4 workloads. Secondly, we can see that SAMR is able to maintain high PM utilization in data centre but the PM utilization of MR method fluctuates, falling down under 80% frequently. This is due to the starvation or unbalanced usage among multiple resource types in MR Thirdly, we observe that the utilization of CPU and RAM resources using SAMR are close in the three synthetic workloads but the difference in Google trace is large. This is caused by the fact that the total demands of RAM is more than that of CPU in traces from Google Cluster. It can also be verified by the higher resource skew ness factors where the skew ness factors in Google trace are much higher than the other three workloads.

We now perform sensitivity studies on major parameters. We investigate the impact of the system parameters including the degree of heterogeneity, delay threshold, the number of VM types and time slot length on the performance of multiple resource usage. For each experiment, we study the impact of varying one parameter while setting other parameters to their default values.

Impacts by workload heterogeneity. We first investigate the performance under different workload distributions with different degrees of heterogeneity. We run four experiments using Growing pattern in this study. In each experiment, the workload consists of only two types of VMs (the amounts of two types of VM are the same) with the same heterogeneity degree. We use <1,1 >+ <1,1 >, <1,4 >+ <4,1 >, <1,8 >+ <8,1 >, and <1,16 >+ <16,1 >in the first, second, third and fourth experiments, respectively. For all the experiments, we keep the total amounts of dominant resource identical in order to compare the impacts of heterogeneity on resource usage. Fig. 8 shows the results using SD, MR and SAMR with different heterogeneity. It can be seen that the

required number of PMs increases as the heterogeneity increases in SD method but the number of PMs required in MR and SAMR falls with the increase of heterogeneity of the workloads. The reason is that large amounts of resources are wasted in SD, while MR and SAMR are capable to provide balanced utilization of resources. This phenomenon again shows the advantage of heterogeneous resource management for serving diversified workloads in IaaS clouds. The advantage becomes more obvious in SAMR which is specifically designed with skew ness avoidance. Impacts by delay threshold. We use a set of delay threshold (minutes): 15,30,60,90,120. We can see from the figure that the number of active PMs in each time slot reduces as we allow higher delay threshold. This is because a larger D value permits more requests in the waiting queue for powering up additional PMs, and thus the cloud system is able to serve more VMs with current active PMs. In practice, cloud providers is able to set an appropriate D to achieve a good balance between quality of service and power consumption.

Impacts by maximum VM capacitiy. we design an experiment on Google trace where the cloud providers offer different maximum VM capacity. For example, a cloud system with the normalized maximum resource m_i offers $(\log_2 m_i \cdot 100 + 1)$ options on resource type-i. We test three maximum resource values 16%,32%,64%, respectively. From the figure we can see that with bigger VMs offered by providers, more PMs are needed to serve the same amount of workloads. The reason is that bigger VMs have higher chance to be delayed when the utilization of resources in the data center is high.

Impacts by time slot length shows the results for varying slot length from 15 minutes to 120 minutes using Google trace. Our heterogeneous resource management allows cloud providers to specify time slot according to their requirements. As shown in the figure, the number of active PMs can be further optimized with smaller time slots. These results suggest that we can obtain better optimization effect if our proposed prediction model and PM provisioning can be executed more frequently. However, the model computation overhead prohibits a time slot being too small.

CONCLUSION

Real world jobs often have different demands on different computing resources. Ignoring the differences in the current homogeneous resource allocation causes resource starvation on one type and wastage on other types. To reduce the monetary costs for users in IaaS clouds and wastage in computing resources for cloud system, this paper first emphasized the need to have a flexible VM offering for VM requests with different resource demands on different resource types. We then proposed a heterogeneous resource allocation approach named skew ness-avoidance multire source (SAMR) allocation. Our answer includes a VM allocation rule to make sure heterogenous workloads area unit allotted befittingly to avoid skew resource utilization in PMs, and a model-based approach to estimate the acceptable .Number of active pms to operate samr. Particularly for our developed markov chain, we showed its relatively low complexity for practical operation and accurate estimation We conducted simulation experiments to test our proposed solution. We compared our solution with the single dimensional method and the multire source method without skew ness consideration. From the comparisons, we found that ignoring heterogeneity in the workloads led to huge wastage in resources. Specifically, by conducting simulation studies with three synthetic workloads and one cloud trace from google, it revealed that our proposed allocation approach that is aware of hete rogenous vms is able to significantly reduce the active pms in data center, by 45% and 11% on average compared with single-dimensional and multi-resource schemes, respectively. We also showed that our solution maintained the allocation delay within the present target.

REFERENCES

[1] S. Genaud and J. Gossa, "Cost-wait trade-offs in client-side re- source provisioning with elastic clouds," in Proc. of 2011 IEEE International Conference on Cloud Computing (CLOUD'10). IEEE, 2011, 1–8.

[2] Michon, J. Gossa, S. Genaud et al., "Free elasticity and free cpu power for scientific workloads on iaas clouds." in ICPADS. Citeseer, 2012, 85–92.

[3] P. Marshall, H. Tufo, and K. Keahey, "Provisioning policies for elastic computing environments," in Proc. of 2012 IEEE 26th Inter- national Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW). IEEE, 2012, 1085– 1094.

[4] L. Wang, J. Zhan, W. Shi, and Y. Liang, "In cloud, can scientific communities benefit from the economies of scale?" IEEE Transac- tions on Parallel and Distributed Systems, 23(2), 2012, 296–303.

[5] R. V. den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost- optimal schedules in hybrid iaas clouds for deadline constrained workloads," in IEEE CLOUD'10, 2010.

[6] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Cost- and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds," in Proc. of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'12). IEEE Computer Society Press, 2012, 22.

[7] K. Deng, J. Song, K. Ren, and A. Iosup, "Exploring portfolio scheduling for long-term execution of scientific workloads in iaas clouds," in Proceedings of International Conference for High Performance Computing, Networking, Storage and Analysis (SC'13). ACM, 2013, 55.

[8] L. A. Barroso and U. Ho¨ lzle, "The case for energy-proportional computing," IEEE computer, 40(12), 2007 33–37.

[9] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," IEEE Transactions on Parallel and Distributed Systems, 2013.

[10] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dy- namic right-sizing for power-proportional data centers," in IN- FOCOM'11, 2011.

[11] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in Proceedings of the Third ACM Symposium on Cloud Computing. ACM, 2012.

[12] L. Wei, B. He, and C. H. Foh, "Towards Multi-Resource physical machine provisioning for IaaS clouds," in IEEE ICC 2014 - Selected Areas in Communications Symposium (ICC'14 SAC), 2014.

[13] D. Villegas, A. Antoniou, S. M. Sadjadi, and A. Iosup, "An analysis of provisioning and allocation policies for infrastructure- as-a-service clouds," in Proc. of 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid'12). IEEE, 2012, 612–619.

[14] K. Mills, J. Filliben, and C. Dabrowski, "Comparing vm- placement algorithms for on-demand clouds," in Proc. of CLOUDCOM'11, 2011.

[15] E. G. Coffman Jr, M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: A survey," in Approximation algorithms for NP-hard problems. PWS Publishing Co., 1996, 46–93.