Efficient Virtual Machine Resource Management for Media Cloud Computing

Mohammad Mehedi Hassan[1], Biao Song[1], Ahmad Almogren[1], M. Shamim Hossain[1], Atif Alamri[1], Mohammed Alnuem[1], Muhammad Mostafa Monowar[2] and M. Anwar Hossain[1]
[1]College of Computer and Information Sciences, King Saudi University
Riyadh, 11543, Kingdom of Saudi Arabia
[e-mail: {mmhassan, bsong, , ahalmogren, mshossain, atif, malnuem, mahossain}@ksu.edu.sa]
[2]Department of Information Technology, Faculty of Computing and Information Technology,
King AbdulAziz University
Jeddah, 21589, Kingdom of Saudi Arabia
[e-mail: hemal.cu@gmail.com]
*Corresponding author: Mohammad Mehedi Hassan

Received October 23, 2013; revised January 4, 2014; accepted March 31, 2014; published May 29, 2014

Abstract

Virtual Machine (VM) resource management is crucial to satisfy the Quality of Service (QoS) demands of various multimedia services in a media cloud platform. To this end, this paper presents a VM resource allocation model that dynamically and optimally utilizes VM resources to satisfy QoS requirements of media-rich cloud services or applications. It additionally maintains high system utilization by avoiding the over-provisioning of VM resources to services or applications. The objective is to 1) minimize the number of physical machines for cost reduction and energy saving; 2) control the processing delay of media services to improve response time; and 3) achieve load balancing or overall utilization of physical resources. The proposed VM allocation is mapped into the multidimensional bin-packing problem, which is NP-complete. To solve this problem, we have designed a Mixed Integer Linear Programming (MILP) model, as well as heuristics for quantitatively optimizing the VM allocation. The simulation results show that our scheme outperforms the existing VM allocation schemes in a media cloud environment, in terms of cost reduction, response time reduction and QoS guarantee.

Keywords: Multimedia cloud, composite media service, VM resource allocation, linear programming and heuristics

This work was supported by NSTIP strategic technologies programs, number (12-INF2613-02) in the Kingdom of Saudi Arabia. We express our thanks to the Editors and anonymous reviewers for their detailed comments and valuable contributions to improve the manuscript.

http://dx.doi.org/10.3837/tiis.2014.05.004

1. Introduction

In recent years, multimedia cloud computing [1] is becoming a promising technology to provide a flexible stack of computing, storage and software services in a scalable and virtualized manner for media-rich applications [2][3][4]. In this media cloud environment, the virtualization technology is applied to package the required CPU, memory, GPU (graphics processing unit), storage and network bandwidth resources of servers into virtual machines to manage and provision heterogeneous multimedia services and applications at lower cost with minimal efforts. These services include but are not limited to image/video retrieval, video transcoding, streaming, video rendering, media analytics, sharing and delivery [5][6][7].

Due to the heterogeneity and mobility of the media services and users, media cloud has brought up the need for an efficient VM resource management to satisfy the QoS requirements of the media services, especially when different atomic media services such as streaming service, video transcoding services, rendering services and so on are composed to meet the customer demands [8]. These services have different QoS requirements, and need dynamic VM resource capacity at the run-time [1]. In addition, the processing delay of both the atomic and composite media services at the server side under different network conditions makes it difficult to efficiently manage VM resources, while fulfilling QoS demands [1][9][10].

Although there are many researches going on to study various VM resource management techniques in cloud environment [11]-[22], very few of them are suitable for media cloud enthronement (Section2 provides a detailed survey of these works related to current paper). This is due to the dynamic nature of the multimedia services (atomic and composite) in terms of variable resource requirements at run-time. Currently, there exist few researches [23]-[28] related to VM resource allocation in a multimedia cloud environment. However, most of them do not take into account the composite media service scenario, which can affect the response time as well as the overall utilization of the physical resources. In addition, they do not consider the multiple VM resource dimensions (i.e. CPU, GPU, memory, storage and network bandwidth) in the resource allocation problem.

In this paper, we tackle the aforementioned challenges of VM resource allocation in a media cloud environment. We propose a VM resource allocation model that optimally allocates VM resources to a set of physical machines/servers by considering the dynamic VM resource requirements for atomic and composite media services. It also ensures the minimum QoS requirements of the multimedia services, while maintaining high system utilization by avoiding over provisioning the VM resources for the services. The proposed VM resource allocation model is designed to 1) minimize the number of physical servers/machines for cost reduction and energy savings; 2) control the processing delay of multimedia services to improve response time; and 3) achieve load balancing or overall utilization of physical resources.

To the best of our knowledge, none of the follwing works [23]-[28] so far presented as a comprehensive approach that handles all the above objectives at the same time with regards to the resource allocation for a media cloud platform. We formulate the proposed VM resource allocation problem into the multidimensional bin-packing problem, which is NP-complete [29]. Similar to this problem, we consider each virtual machine as an item, and the dimen- sions as its capacities. The goal is to minimize the number of physical servers

to be used to place all Virtual Machines, while considering physical servers' capacities. To reduce the response time, improve the overall resource utilization and avoid frequent VM migration, we additionally consider three constraints such as processing delay, overall resource utilization and special resource utilization. We further define a mechanism that dynamically determines the optimal threshold value of these constraints. Based on the above considerations, we design a MILP model for optimizing VM allocation into physical servers. We also use various heuristics to generate candidate resource allocation schemes and to choose the best scheme for real VM allocation.

Several experiments were carried out to validate the efficiency of our pro- posed VM resource allocation model in media cloud platform. These experiments were conducted for different request patterns of media services in various environments. We have also compared our proposed algorithm with three other existing algorithms in media cloud platform, which comprised of a load balancing model [28], queuing model [25], and a round-robin allocation [21]. The results include the performance of cost reduction, response time reduction and a QoS guarantee.

The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 describes our VM resource allocation model and heuristics for the current multi-dimensional problem. Section 4 presents experimental results and performance comparisons. Finally, Section 5 concludes the paper.

2. Related Work

In recent years, the VM resource allocation in cloud environment has gained significant attention. Many existing efforts [11]-[22] study various VM resource management techniques for cloud resource management. Beloglazov et al [11] investigated energy-aware VM resource allocation heuristics that provides data center resources to client applications in a way that improves the energy efficiency of the data center without violating the negotiated SLAs. Aisopos et al [15] proposed a VM resource allocation model for SaaS cloud providers using fractional knapsack problem that maximizes the service provider's revenue and the resource utilization under a heavy load. The model focuses on maintaining the maximum resource utilization at the cost of risking potential SLA violations over the pending jobs that will yield the smaller profit for the SaaS cloud provider. Lin et al [17] developed a self-organizing model to manage cloud VM resources without the requirement of a centralized management control. Teng et al [18] presented a resource pricing and equilibrium allocation policy for limited cloud user competition over resources. Van et al [16] presented an autonomic virtual resource management mechanism in a cloud for service hosting platforms. Similarly, Berral et al [21] focused on autonomic energy-aware scheduling in the cloud that dynamically adapts to varying task types and workloads, and even to varying cloud infrastructure. The emphasis there on replacing uncertain information with data mining based algorithms, Game-theory based VM resource management for clouds was also studied in [19][20].

However, the works presented above do not consider media cloud environment scenario, where a short battery lifetime, varying wireless channel conditions, and interaction latency pose major challenges for VM resource allocation. Additionally, these VM resource allocation models are unsuitable for heterogeneous multimedia services (atomic and composite), which are basically dynamic in nature. Currently, there are some research efforts going on [23]-[28] for allocating VM resources in a multimedia cloud environment. Sembiring

et al. [23] investigated the relationship between the properties of media tasks and their resource consumption and accordingly propose a dynamic resource allocation solution utilizing machine learning techniques. Nan et al [25] proposed a cost-effective resource allocation optimization approach for multimedia cloud that was based on a queuing model. They considered the data center infrastructure as a node-weighted tree-like graph, and then used the queuing model to capture the relationship between the service response time and the allocated resources. They also studied the resource allocation problem in a single-class service case and a multiple-class service case, respectively. The same authors also present similar approach in [24][26][27].

Wen et al [28] presented an effective load-balancing algorithm for a cloud- based multimedia system, which can allocate and schedule VM resources for different user requests with minimum costs. They considered the network proximity and node server traffic loads while making an allocation. They utilized a round-robin algorithm to find the optimal node servers, which would minimize costs. However, those who work in the multimedia cloud assume that the pool of VM resources is homogeneous and that all multimedia service tasks are atomic; this is not practical. In [6], Miao et al presented a cloud-based free viewpoint video rendering framework. In this architecture, every time, the user requests a new viewpoint, the view will be rendered in the cloud and then sent to the client. The researchers also proposed a resource allocation scheme that jointly considered rendering and rate allocations between the cloud and client to optimize the QoE (Quality of Experience). Here, the resource allocation is restricted to the free viewpoint video rendering framework and cannot be used for the general media cloud environment.

Some research [30][31] focused on VM allocation for virtual desktop cloud (VDC) environment from the thin client perspective. In [30], Calyam et al. presents a utility-directed resource allocation model (U-RAM) that uses an offline benchmarking based combined utility function of CPU, memory, net- work health and thin-client user QoE. This is to dynamically create and place virtual desktops in resource pools at distributed cloud data centers, while optimizing resource allocations. This is accomplished by considering timeliness and coding efficiency as quality dimensions. They also propose an iterative algorithm for U-RAM (an NP-Hard problem) to optimize resource allocation with fast convergence that is based on combined utility functions. However, the U-RAM model cannot be directly applicable in the media cloud environment since it does not consider the composite multimedia service or application. This can affect the response time as well as overall utilization of VM resources.

There are some others works which focus on resource allocation in heterogeneous multimedia services and network. For example, in [38], the authors presented a framework for efficient bandwidth allocation and group cooperation in P2P Live streaming scneario. In [39], the authors developed a fair resource allocaiton algorithm based on game theory for wirless multimedia communications. A non-intrusive and adaptable resource management framework, for multimedia applications, distributed in open and heterogeneous home networks is proposed in [40]

To the best of our knowledge, only a few attempts towards proposing an all-inclusive approach that demonstrates a cost-effective and dynamic VM resource allocation for a media cloud platform, which not only considers atomic as well as composite multimedia services but also heterogeneous net- work conditions along with the overall utilization of physical servers. There- fore, the exploration of such a comprehensive approach is timely and crucial for considering the proliferation of emerging multimedia cloud computing.

3. Proposed VM Resource Allocation Model in Media Cloud Systems

3.1 Problem Formulation

The proposed VM resource allocation problem is mapped to the multidimensional bin-packing problem [29], which is NP complete. In this problem, we have to map several items into the smallest number of bins as possible. Here, each item denotes a tuple, which contains its dimensions. In our scenario, we consider each VM as an item and the dimensions like CPU, memory, storage, network bandwidth, and GPU, as its capacities. The target is to find out the minimum number of physical servers to place all the VMs, with respect to physical server's capacities. To solve this problem, we have designed a MILP model, as well as heuristics to quantitatively optimize the VM allocation. We also consider three additional constraints: the processing delay of media service, resource utilization, and the special resource utilization to reduce the response time, improve the overall resource utilization and avoid frequent VM migration respectively.

The VM resource allocation process can be static or dynamic. In static VM allocation, VM capacities are configured using peak load demands of each media workload. The utilization of the peak load demand ensures that the VM does not overload and stay in the same physical servers during their entire lifetime. However, it leads to idleness due to the variable VM resource demand of media workloads.

In dynamic VM allocation, VM capacities are configured dynamically according to the current media workload demands. However, it may require migrating VMs between physical servers in order to: (i) pull out physical servers from an overloaded state when the sum of VMs capacities mapped to a physical server becomes higher than its capacity; (ii) turn off a physical server when the VMs mapped to it can be moved to other physical servers. In our scenario, we allow the virtual machine capacities to be varied on demand. When new media service joins, the system uses the proposed VM allocation algorithm to find proper physical server. If VM migration is required due to the occurrence of an overloaded state, the system also adopts the same VM allocation algorithm for selecting a new physical server. Now, we present a linear programming model to solve the VM resource allocation problem.

3.2 Linear Programming Model

The input parameters and variables used in the linear programming formulation are presented in **Table 1.** For an atomic or composite media service I that needs to be allocated, the MILP model is presented in Eq. (1) to (6).

$$\text{Minimize} \quad \sum_{p \in P} y_p \tag{1}$$

Subject to

$$\sum_{p \in P} x_{pv} = 1 \quad \forall v \in V \tag{2}$$

$$\sum_{v \in V} u_{vr} x_{pv} \le c_{pr} y_p \quad \forall p \in P, \forall r \in R \tag{3}$$

$$d_I \le T \tag{4}$$

$$\sum_{v \in V} ruc_{pv} x_{pv} \le T_1 \quad \forall p \in P \tag{5}$$

$$\sum_{v \in V} s_{pv} x_{pv} \le T_2 \quad \forall p \in P \tag{6}$$

The objective function in (1) aims at minimizing the number of required physical servers. The constraint in (2) guarantees that each virtual machine is mapped to a single physical server. Equation (3) guarantees that the virtual machine demands allocated in each physical server do not overload its capacity. The constraint in (4) guarantees that the processing delay of media service *I* does not exceed a certain threshold value *T*. Equation (5) is an optional constraint which can help to improve the overall resource utilization. The optional constraint in (6) can reduce the chance of special resource overload and can potentially balance the special resource utilization among all physical servers. Here special resource means that some media applications may give more importance to only CPU than other resources or combinations of any resources. In this scenario, the constraint in (6) cans address to balance the special resource utilization throughout the physical servers.

Since the future workload may not be predictable, our objective function in (1) represents the average statistics from time 0 to current time. For the constraints, they should be satisfied at any time when the allocation decision is made. Thus, we did not present the time variable t in those formulas. The optional constraints (5) and (6) are proposed to reduce the searching space for this NP-hard optimization problem. However, the use of these constrains may lead the results to be near-optimal. The definitions and effectiveness of ruc_{pv}, S_{pv} and d_I are explained as follows:

Table 1. Parameters and variables for the VM allocation problem

Parameters	Description
P	set of physical servers
V	set of virtual machines
R	set of resources (CPU, memory, storage, GPU and network)
$I = \{v_1, v_2, ..., v_n\}$	multimedia service (single or composite)
u_{vr}, $v \in R \,\&\, r \in R$	amount of resource *r* used by VM *v*
$c_{pr} \in R$	capacity for physical server $p \in P$ of resource $r \in R$
ruc_{pv}	overall resource utilization on physical server $p \in P$ after allocating virtual machine $v \in V$
S_{pv}	percentage of special resource utilization on physical server $p \in P$ after allocating virtual machine $v \in V$
$d_{p_1 p_2}$	delay between physical servers $p_1 \in P$ and $p_2 \in P$
d_{pe}	delay between physical server $p \in P$ and external server *e*
d_I	delay of media service *I*
$y_p \in \{0,1\}$	equals to 1 if physical server $p \in P$ is used, 0 otherwise
$x_{pv} \in \{0,1\}$	equals to 1 if virtual machine $v \in V$ is allocated to physical server $p \in P$ currently, 0 otherwise

Definition 1: (media service delay): Given a multimedia service $I=\{v_1,v_2,...,v_n\}$, we discuss three types of service flow. The first case is for the atomic multimedia service case, where the VMs have no intercommunication with each other. In this case, the constraint on d_I can be defined as follows:

$$\sum_{p\in P} d_{pe_i} x_{pv_i} \le T_{v_i} \quad \forall v_i \in I \tag{7}$$

We apply different threshold values for different Virtual Machines (T_{v_i} for v_i). For any virtual machine v_j that does not have communication with an external server, the corresponding constraint can be removed. The second scenario is for the composite multimedia service with synchronous integration, where the screen update from n Virtual Machines is synchronized first, and is then transmitted to the user. In this case, the constraint on d_I can be denoted as follows:

$$\sum_{p\in P} d_{pe_i} x_{pv_i} \le \min(T_{v_i}) \quad \forall v_i \in I \tag{8}$$

Where $\min(T_{v_i})$ represents the most strict delay constraint among all of the services. Every virtual machine allocation should meet the requirement of $\min(T_{v_i})$, since they are synchronized.

The last case is for the composite multimedia service with sequential composition, where the output of the service running on v_i is the input of the service running on v_{i+1}. In this case, the constraint on d_I can be specified as follows:

$$\begin{aligned} & \sum_{p\in P} d_{pe_1} x_{pv_i} \le T_{v_1} \\ \textit{Given } p_1 \quad & \sum_{p\in P} (d_{pe_2}+d_{p_1 p}) x_{pv_2} \le T_{v_2} - T_{v_1} \\ & \cdots \\ \textit{Given } p_{n-1} \quad & \sum_{p\in P} (d_{pe_n} + d_{p_{n-1} p}) x_{pv_n} \le T_{v_n} - T_{v_{n-1}} \end{aligned} \tag{9}$$

In this case, the delay constraint on allocation is computed by subtracting T_v from T_{v+1}.

Definition 2 (Resource utilization and Special Resource condition): Given a media service or application i and a physical machine j, let c_{ij}, m_{ij}, g_{ij}, and b_{ij} are the percentages of resource usage regarding CPU, memory, GPU and network bandwidth, respectively. If the application is supported by the VM, then the extra resource consumption of VM, OS and remote server should be also included. Thus, application i and virtual machine v can be used interchangeably. Let fc_j, fm_j, fg_j, and fb_j be the percentages of idle CPU, memory, GPU and network bandwidth resources, respectively, on machine j. For any resource, if at least k % of free capacity is reserved to buffer the unexpected workload burst, then it would

not be counted in the available idle resources. The exact amount of reserved resource is determined by the cloud resource provider by using long term benchmark or short term workload prediction models [32][33]. The media application can be allocated to that physical machine only if the following condition (10) is met:

$$c_{ij} \le fc_j \;\&\; m_{ij} \le fm_j \;\&\; g_{ij} \le fg_j \;\&\; b_{ij} \le fb_j \tag{10}$$

After media application i is allocated on physical machine j, the average percentage of free resource ap_{ij} is defined using (11):

$$ap_{ij} = (fc'_j + fm'_j + fg'_j + fb'_j)/4 \tag{11}$$

where,

$$\begin{aligned} fc'_j &= fc_j - c_{ij},\ fm'_j = fm_j - m_{ij}, \\ fg'_j &= fg_j - g_{ij},\ fb'_j = fb_j - b_{ij} \end{aligned} \tag{12}$$

For machine j, the resource utilization condition after allocating application i is denoted as ruc_{ij}, which is a mean-square value. Using (13), we have defined ruc_{ij} in (5) as follows:

$$ruc_{ij} = (fc'_j - ap_{ij})^2 + (fm'_j - ap_{ij})^2 + (fg'_j - ap_{ij})^2 + (fb'_j - ap_{ij})^2 \tag{13}$$

We assume that the applications running on a single physical machine share all of the resource capacity in a proportional way. More specifically, application i will be allocated on physical machine j, which will get $(c_{ij}/(1-fc'_j)) \times 100\%$ of the CPU capacity, and same for memory, GPU and bandwidth resoueces.

As the resources may not be treated equally in multimedia system, constraint (6) is created to address this issue. Let us consider an instance where CPU is more important for processing the multimedia tasks, such as face recognition or data analytics. Regarding application i, the superiority of machine j is defined as s_{ij} in (6) by using (14):

$$s_{ij} = c_{ij}/(c_{ij}/(1-fc'_j)) = 1 - fc'_j \tag{14}$$

It turns out that for allocation i the superiority of machine j is the percentage of CPU capacity that has been occupied after application i's allocation. For other systems where bandwidth or GPU is more important, the definition of s_{ij} can be modified accordingly.

3.3. Threshold Determination

In our MILP formulation, we have three types of threshold: network delay T in (4), overall resource condition T_1 in (5), and free resource T_2 in (6). The delay threshold T is given by the QoS requirement of each application. Certain amount of special resource capacity should be reserved to handle an unexpected resource burst. Thus, T_2 will be generated according to the specific QoS requirement, the benchmark or the workload burst prediction regarding each

application. The applications that experience frequent special resource bursts may require a small T_2 value.

Normally, we use the objective function presented in (1) to determine the optimal value of T_1. However, if the physical servers are not capable of supporting all of the requests, in order to find the optimal value of T_1, we will introduce a new objective function U using (15):

$$U = \lambda_c[\sum_{p_j \in P}(1-fc_j)\times oc_j] + \lambda_m[\sum_{p_j \in P}(1-fm_j)\times om_j] + \lambda_g[\sum_{p_j \in P}(1-fg_j)\times og_j] + \lambda_b[\sum_{p_j \in P}(1-fb_j)\times ob_j] - C_{P'} \quad (15)$$

where U is the overall resource utilization; oc_j, om_j, og_j, and ob_j are the total amount of CPU, memory, GPU and network bandwidth on physical server p_j; λ_c, λ_m, λ_g and λ_b denote the pricing schemes for each CPU unit, memory unit, GPU unit and network bandwidth unit, respectively; $C_{P'}$ is the total cost for maintaining the active physical servers. This new objective function measures the total server resource utilization. For the simplicity of explanation, we describe the threshold selection algorithm with the objective function U. It is exactly same as if the objective function U is replaced by the one that we presented in (1). Suppose that we have a set of applications, a set of physical servers and a heuristic adopting threshold T_1. By varying the value of threshold and by repeatedly using the heuristics, we can get different results regarding the overall resource utilization U.

Let $f(T_1) = U\ (0 <= T_1 <= 1)$ be the evaluation function. Our goal is to find an optimal T_1, which can maximize U. Intuitively, the function $f(T_1)$ should be nearly unimodal in its domain. Considering the optimal T_1 as the standard point, a lower T_1 will block proper VM allocation while a higher T_1 may cause improper VM allocation. Both cases result in low resource utilization since more physical servers have to be used to handle VM requests.

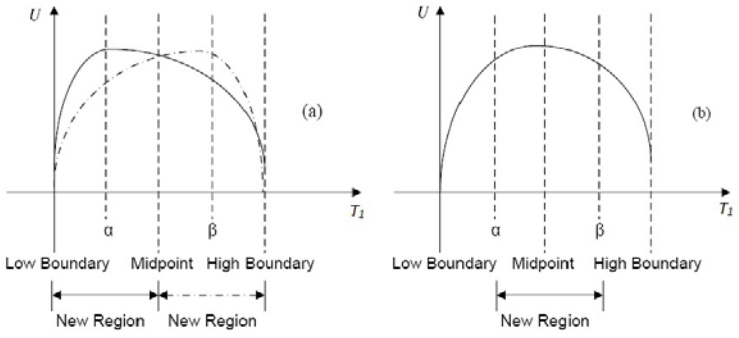

Fig. 1. The rules for choosing threshold value (a). The peak is located in the side part (b). The peak is located in the middle part.

We now propose an iterative approach to determine the optimal value of T_1. The principle of our approach is very similar to that of Bisection method [34]. We repeatedly use the

heuristics to simulate the VM allocation process, where different T_1 are used. According to the results of U, we bisects the original interval then selects a subinterval where a T_1 maximizing $f(T_1)$ must lie for further processing. The updating rule is shown in **Fig. 1**.

In **Fig. 1**, α and β are the midpoints between the current three points. Fig. 1(a) shows the situation where the peak point is located in the side part. If $f(\alpha) > f(Mid)$, the current midpoint is set as the new high boundary, while α is set as the new midpoint. Similarly, if $f(\beta) > f(Mid)$, the current midpoint is set as the new low boundary, while β is set as the new midpoint. As shown in Fig. 1(b), for $f(\alpha) > f(Mid)$ and $f(\beta) > f(Mid)$, α is set as the new low boundary, while β is set as the new high boundary.

According to the definition domain of T_1, the low boundary is initialized to be $T_1 = 0$, and the high boundary is $T_1 = 1$. Therefore, the medium point between the low boundary and the high boundary is $T_1 = 0.5$. These three points are iteratively updated until the interval between the low boundary and the high boundary is small enough. It is not necessary to determine T_1 for each set of VM request during runtime. Only a periodical determination is required.

3.4. Heuristics to Model the VM resource Allocation

The multidimensional bin-packing problem can also be solved using heuristics. Although heuristic solutions will not guarantee an optimal solution, the required time to obtain a feasible solution is much shorter than MILP. We utilize three heuristics like first-fit decreasing (FFD), best-fit decreasing (BFD), and worst fit-decreasing (WFD) [29] and modify them according to the restrictions of the constraints (eq. (1) - (6)). In that heuristics, a lexicographic order is utilized to sort each VM demand. Following the heuristic definitions, the mapping of each VM will then be performed. In the FFD heuristic, the VM will be mapped to the first physical server with available capacity. In case of a BFD heuristic, it will be mapped to the physical server that leaves the least left over space after the mapping between all available physical servers. And for WFD heuristic, it will be mapped to the physical server that leaves the largest left-over space after the mapping between all available physical servers. We adopt these three heuristics to generate candidate VM allocation schemes, and to choose the best scheme for real VM allocation.

3.4.1. Algorithm Description

In this section, we explain the entire procedure of our proposed allocation algorithm. **Algorithm 1** presents the pseudo code of the proposed allocation algorithm.

***Step 1*:** Check whether it is necessary to re-select the threshold value. If it is necessary, then run the threshold selection algorithm; Otherwise, use the previous threshold value.
***Step 2*:** Use the three heuristics FFD, BFD and WFD with the threshold value to generate three allocation schemes.
***Step 3*:** Choose the best allocation scheme and enforce allocation.
***Step 4*:** Update the physical server information for next allocation.

3.4.2. Complexity Analysis

The time complexity of the proposed allocation algorithm is denoted by $O(3\times n^2m+9\times\log_2 100\times pn^2m)$. n denotes the number of service request, m denotes the number of physical servers, and p denotes the probability that the threshold will need to be re-selected. The time complexity of heuristic allocation algorithm is $O(3\times n^2m)$, since it calls FD, BFD and WFD functions where each function takes $O(nm)$ to allocate one service request, and $O(n^2m)$ for the entire allocation. The threshold determination algorithm consists of $\log_2 100$ rounds, and heuristic allocation algorithm is called three times in each round. Thus, each execution of threshold determination algorithm takes $O(9\times\log_2 100\times n^2m)$. As it is executed with probability p, the total complexity of threshold determination part is $O(9\times\log_2 100\times pn^2m)$.

Algorithm 1. Pseudo code of VM resource allocation algorithm

```
if is_time_to_update_threshold() then
    T_low ← 0, T_high ← 1, T_medium ← 0.5;
    while (T_high − T_low>0.01) do
        T_α ← (T_low + T_medium)/2;
        T_β ← (T_high + T_medium)/2;
        if heuristic(T_medium) ≥ heuristic(T_α) ≥
        heuristic(T_α)orheuristic(T_medium) ≥ heuristic(T_β) ≥ heuristic(T_α)
        then
            T_low ← T_α;
            T_high ← T_β;
        else
            if heuristic(T_β) ≥ heuristic(T_medium) ≥ heuristic(T_α) then
                T_low ← T_medium;
                T_medium ← T_β;
            end
        end
    end
end
T_1 ← T_medium;
heuristic(T_1);
enforce_allocation();
update_server();

heuristic (threshold T_1) {
best_result ← φ;
if heuristic_FFD(T_1) > best_result then
    best_result ← heuristic_FFD(T_1);
end
if heuristic_BFD(T_1) > best_result then
    best_result ← heuristic_BFD(T_1);
end
if heuristic_WFD(T_1) > best_result then
    best_result ← heuristic_WFD(T_1);
end
return best_result }
```

4. Performance Evaluation

In this section, we have presented simulation setup description and conducted several experiments to validate the efficiency of our proposed VM allocation approach as done in. These experiments were conducted for different cases, such as low/high heterogeneity of media tasks, and a large/small media task set. We compared our proposed algorithm with three existing algorithms: a load balancing model [28], a queuing model[25], and a round-robin allocation [21]. The results include the performance of cost reduction; response time reduction and QoS guarantee.

4.1. Simulation Settings

In order to describe the simulation setup in a clear way, we draw the entire simulation infrastructure and present it in **Fig. 2**. As we can see from the figure, there are two major simulation components: workload generator and cloud simulator. The workload generator takes responsibility of generating atomic, synchronous and sequential workload for simulation. The two sub-components are designed to generate individual multimedia service workload and to generate delay constraints for three types of service composition, respectively. The cloud simulator receives multimedia service requests from workload generator and creates VMs with pre-configured CPU capability, memory, GPU and bandwidth. The cloud resourcesand network environment are simulated by generating physical machines/servers with identical capacity and network latency matrix indicating the network latency between the physical machines/servers, respectively. Finally, seven different resource allocation schemes, including proposed method are tested. The results such as the number of active physical machine/server and average delay are collected from PM (physical machine) monitor and network monitor.

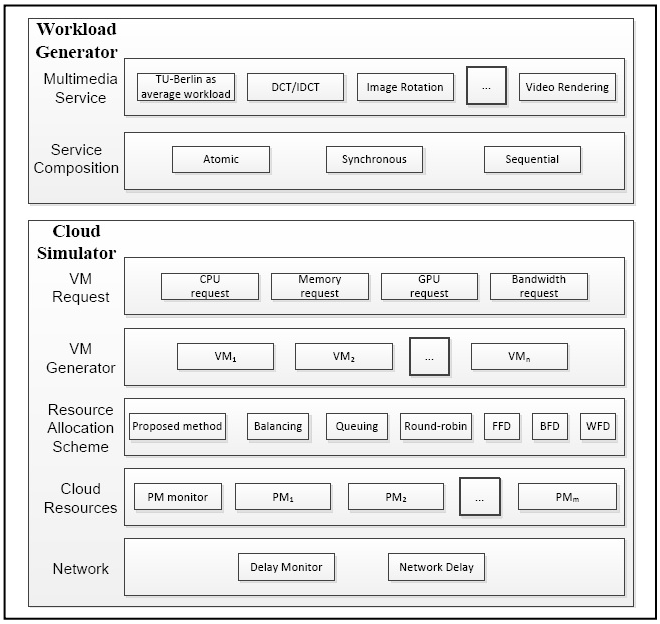

Fig. 2. Silumation Infrstructure

Table 2 shows simulation parameters, where *HI*, *ACU*, *AMU*, *APU*, *ANU* represents heterogeneity index, average CPU utilization(%), average memory utilization(%), average GPU utilization(%), and the average network bandwidth utilization(%) respectively. *HI*, *ACU* and *AMU*were retrieved from the Technical University of Berlin (TU-Berlin) workload [8], which are normally used by researchers and students to execute computational experiments. redTo address multimedia service issues in our simulation, we generate the workload by considering several representative multimedia service cases. The first case is Discrete Cosine Transform/Inverse Discrete Cosine Transform (DCT/IDCT), which is very CPU intensive and mostly used in the MPEG and JPEG encoding/decoding [35][36]. Secondly, image rotation has relatively low demand on CPU capacity, but needs more memory and bandwidth. We also consider video rendering workload withhigh demand on GPU capability. As the size of image/video can be different between the service instances, the workload should not be generated in an identical way even for the same multimedia service case. Using the above simulation parameters and the patterns of multimedia service, we randomly generate multimedia service requests in each workload. Initially, the capacities of physical servers were assumed to be identical. In this simulation, the number of physical server was fixed to 100.

Table 3 specifies the delay settings, where *IDC*, *SDC*, *IDT* and *SDT* represents the individual delay constraint on atomic media service, the sequential delay constraint on adjacent media services, the individual delay time on a single server, and the sequential delay time on connected servers respectively. The variation of heterogeneity only affects the delay constraint on media services.

While allocating the services, two types of allocation schemes are adopted. In large group allocation case, all media service requirements are assumed to be generated and submitted at one time. On the other hand, small group allocation allows only 1-5 media service requests submission. When the current allocation is done, the following 1-5 media service requests will be created. In both cases, each group of composite media services contains 1-5 services, which can be atomic, synchronous or sequential. Theservice composition does not introduce more resource consumption, but can change the delay constraint on the services.

Table 2. Details of workload group

Heterogeneity	Number of traces	HI	ACU	AMU	AGU	ANU
Low	50-200	0.17	25.2%	28.36%	30%	30%
High	50-200	063	47.28%	48.67%	50%	50%

Table 3. Details of delay

Heterogeneity	IDC	SDC	IDT	SDT
Low	25ms-35ms	25ms-35ms	5ms-30ms	5ms-15ms
High	15ms-45ms	25ms-45ms	5ms-30ms	5ms-15ms

4.2. Experiments

4.2.1. Cost Optimization

Several sets of experiments were conducted in the simulation. Firstly, we adopted different request patterns of media services/applications in the experiment (i.e. large/small media service group at Low/high heterogeneous environment) to measure the cost optimization capability. The number of media requests was fixed to 100.

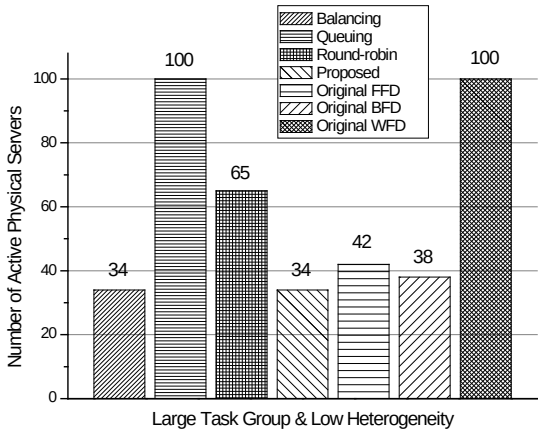

Fig. 3. Cost optimization in large task group at low heterogeneity environmen

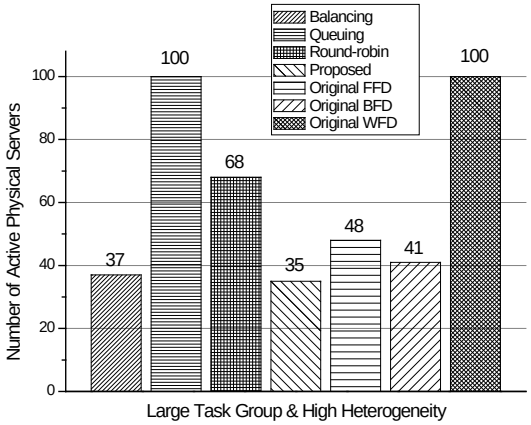

Fig. 4. Cost optimization in large task group at high heterogeneity environment

Fig. 3 shows the simulation results derived from the large task group at low heterogeneity environment. From the results, we have found that our proposed approach and the load balancing method have the same performance. The optimal allocation sequence is applied by the load balancing method as well as our proposed approach. Since the low heterogeneity requests do not overuse any type of resource, the resource utilization condition threshold we used in our proposed approach does not provide further optimization in this environment. The queuing model performs worse than any other solution as it does not consider virtual machine. Thus, each service request must occupy one physical server. The round-robin method randomly chooses a physical server for each request. Since it does not provide any optimization method, we have found that 65 physical servers need to be launched according to the allocation results of round-robin method. redThe original FFD, BFD and WFD are also tested in this simulation environment. The results suggest that those original heuristics considering one dimension (CPU capacity) have inefficient performance while dealing with the cloud resource allocation problem on multimedia service. The original WFD, since it always puts VM on a physical server with highest CPU capacity, activates 100 physical servers to handle 100 service requests. The allocation result of original FFD and that of original BFD cause 42 physical servers and 38 physical servers to be active, respectively.

In **Fig. 4**, we present the results retrieved from the large task group and high heterogeneity environment. Due to the resource utilization condition threshold, our proposed approach outperforms existing algorithms by avoiding overuse of any resource. The performance of load balancing method does not degrade so much, as allocating a large group of requests is easier than allocating several small groups of requests. While allocating a large group of requests the load balancing method can always choose the best allocation among all the possible allocations. As the requests consume more resources, the results of round-robin, original FFD and original BFD increases to 68, 48 and 41 active servers, respectively. The performance of queuing method and that of orignal WFD remains same as what we present in **Fig. 3** due to the reasons we explained before.

It can be seen from **Fig. 5** that our proposed also achieves best performance in the small task group and low heterogeneity environment. The advantage of our proposed approach was greatly amplified in the high heterogeneity environment, which is more similar to the real scenario. The results are presented in **Fig. 6**. Compared with the load balancing method, more

than a 10% improvement is achieved from using our proposed approach. Thus, we can conclude that our proposed approach is more suitable for the real multimedia service allocation.

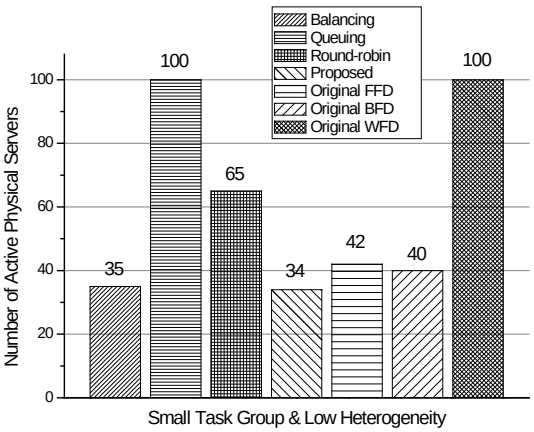

Fig. 5. Cost optimization in small task group at low heterogeneity environment

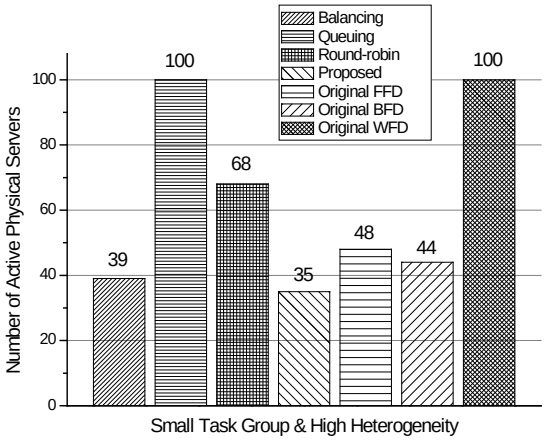

Fig. 6. Cost optimization in small task group at high heterogeneity environment

4.2.2. Scalability

In order to validate the scalability of our proposed approach, we varied the number of service requests to test the cost in the small task group and a high heterogeneity environment. **Fig. 7** shows the results of the scalability test. When the total number of service requests equaled 50, the performance of our proposed approach was slightly better than that of the load balancing method. By increasing the workload, a significant difference can be found in the graph. Compared with the loadbalancing method, our proposed approach demonstrates better scalability. The queuing method and the round-robin method have obvious drawbacks regarding scalability.

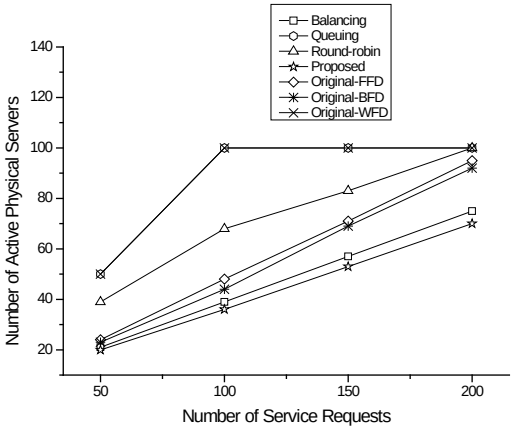

Fig. 7. Scalability Test Results

The queuing method can support maximum 100 service requests with 100 physical servers. extcolorredThe similar result is observed from the original WFD algorithm except that more than 100 service requests are actually allocated due to the use of VM. Conversely, the

round-robin algorithm cannot allocate more than 200 service requests on 100 physical servers. The original FFD and BFD algorithms perform better than round-robin algorithm, but worse than proposed method and balancing method.

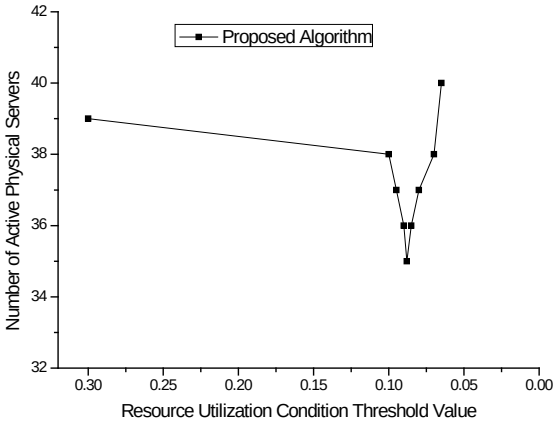

Fig. 8. Resource utilization condition threshold vs. cost

In the previous simulations, we allowed our proposed approach to use the optimal resource utilization condition threshold value. It was also necessary to show the importance of the threshold selection. We adopted a different threshold value in our proposed approach and presented the cost optimization results in **Fig. 8**. From the graph, we can see that our proposed approach can achieve the best performance when the selection of the threshold value is appropriate. Otherwise, the performance may significantly degrade. The results validate the importance of our threshold selection algorithm.

4.2.3. Response time and QoS Success Rate

In the second set of simulations, we explore the actual response time and QoS success rate in different environments. redSince the original FFD, BFD and WFD algorithms do not consider delay issue as the round-robin algorithm does, we do not present similar results of those algorithms. In **Fig. 9** and **Fig. 10**, we present the outcomes that we recorded in a low heterogeneity environment.

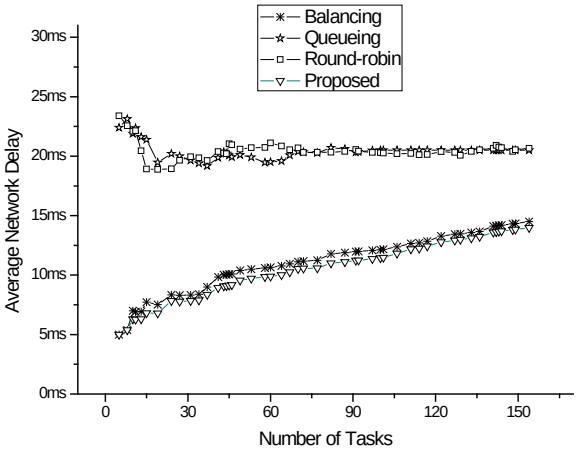

Fig. 9. The response time in low heterogeneity environment

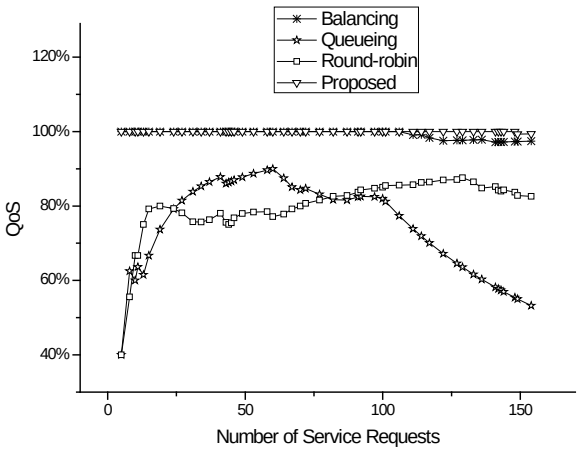

Fig. 10. The QoS success rate in low heterogeneity environment

In **Fig. 9**, the average delay (response time) of the services achieved by each solution is illustrated. The reason why the queuing and round-robin performs is worse than the load balancing and our proposed approach is because they do not consider the delay optimization during the VM allocation process. The loadbalancing method is able to optimize atomic delay for each service allocation. Our proposed approach performs better than the load balancing approach since we clearly define the delay model for both atomic and composite media services. Thus, we are capable to address the dependency issue among the services and the physical machines.

As shown in **Fig. 10**, our algorithm also achieves a higher QoS success rate. The results of the queuing model decrease dramatically after 100 service requests, as it can only support 100 services with a 100 physical server. The additional service requests must wait in the queue, which means that there is a QoS violation. The round-robin approach maintains an average 70%-80% success rate with random server selection. The load balancing approach cannot meet the QoS requirement when the numberof service requests exceeds 100. Our approach improves this number to 150. At the beginning stage, the composite QoS requirement can be fulfilled by the load balancing approach, since the physical servers with very low individual delay are selected. However, the dependency of services/servers must be considered when the workload increases to a certain level. At that time, the physical servers with very low individual delay have been occupied. Thus, the load balancing approach is more likely to violate QoS requirement without exploring the dependency of services/servers.

Fig. 11 and **Fig. 12** show the results in a high heterogeneity environment. From **Fig. 11**, we can see that the performance of queuing and of the round-robin do not change, since their allocation results are atomic to the service delay requirement. Therefore, the actual response time remains same, as the services are still allocated to the same physical servers. As can be seen from **Fig.12** that the QoS success rate dramatically decreases if we use the queuing allocation or the round-robin allocation. The performance of the load balancing method also has an obvious degradation. Our proposed method shows its superiority, including low response time and high QoS success rate in the high heterogeneity environment. Consequently, we can conclude that our delay model is effective in the composite service allocation.

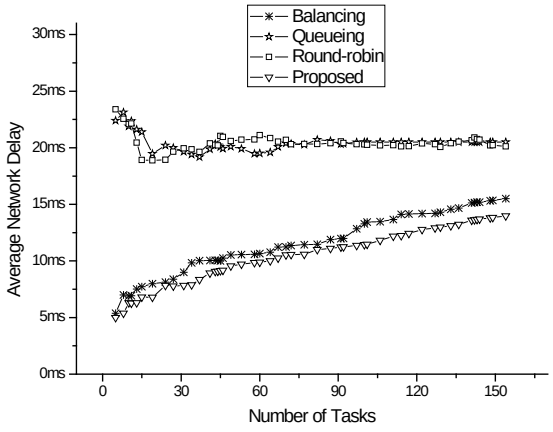

Fig. 11. The response time in high heterogeneity environment

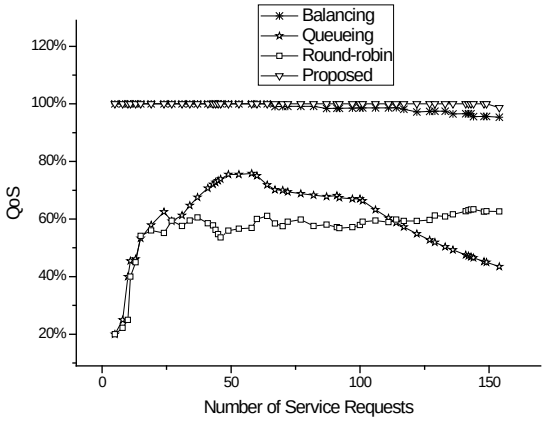

Fig. 12. The QoS success rate in high heterogeneity environment

5. Conclusion

The media cloud is emerging as a remarkable technology that can facilitate effective processing of complex multimedia services and provide QoS provisioning for multimedia service or applications from anywhere, anytime and at any device at lower costs. One major challenge for a cloud provider in such media cloud environment is to find an efficient VM resource allocation model for processing media service tasks. This paper presents a VM resource allocation model that dynamically utilizes VM resources to satisfy QoS requirements of media- rich mobile cloud services or applications. In order to do VM resource allocation effectively, we have presented a MILP model, as well as heuristics. Performance comparisons show that our resource management/allocation approach performs very competitively while satisfying users' QoS demand. This work does not include QoE, media play-back quality, media service profiling and benchmarking. As for the future works, we would incorporate some of above as a part of the future work. We believe that our proposed allocation approach can adapt such settings.

References

[1] Zhu, W., Luo, C., Wang, J., Li, S., "Multimedia cloud computing. Signal Processing Magazine," *IEEE*, 28(3), 59 –69, 2011. Article (CrossRef Link)

[2] Amreen, K., Kamal, K., "Mobile cloud computing as a future of mobile multimedia database," *International Journal of Computer Science and Communication*, vol. 2, pp. 29–221, 2011.

[3] Kumar, K., Lu, Y.H., "Cloud computing for mobile users: Can offloading computation save energy?," *Computer*, 43(4), 51–56, 2010. Article (CrossRef Link)

[4] Simoens, P., De Turck, F., Dhoedt, B., Demeester, P., "Remote display solutions for mobile cloud computing," *Computer*, 44(8), 46 –53, 2011. Article (CrossRef Link)

[5] Dey, S., "Cloud mobile media: Opportunities, challenges, and directions," in *Proc. of International Conference on Computing Networking and Communications (ICNC) 2012*, pp. 929 –933, 2012. Article (CrossRef Link)

[6] Miao, D., Zhu, W., Luo, C., Chen, C.W., "Resource allocation for cloud-based free viewpoint video rendering for mobile phones," in *Proc. of Proceedings of the 19th ACM international conference on Multimedia*, MM '11, ACM, New York, USA, pp. 1237–1240, 2011. Article (CrossRef Link)

[7] Shi, S., Jeon, W.J., Nahrstedt, K., "Campbell, R.H.: Real-time remote rendering of 3d video for mobile devices. in *Proc. of Proceedings of the 17th ACM international conference on Multimedia*, MM '09, ACM, New York, NY, USA, pp. 391–400, 2009. Article (CrossRef Link)

[8] Lin, Q., Tretter, D., Liu, J., O'Brien-Strain, E., "Multimedia analysis and composition cloud service," in *Proc. of* Proceedings of the Third International Conference on Internet Multimedia Computing and Service, ICIMCS '11, ACM, New York, NY, USA, pp. 55–58, 2011. Article (CrossRef Link)

[9] Satyanarayanan, M., Bahl, P., Caceres, R., Davies, N., "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, 8(4), 14–23 (2009). Article (CrossRef Link)

[10] Tolia, N., Andersen, D., Satyanarayanan, M., "Quantifying interactive user experience on thin clients," *Computer*, 39(3), pp. 46 – 52, 2006. Article (CrossRef Link)

[11] Beloglazov, A., Abawajy, J., Buyya, R., "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, 2011. Article (CrossRef Link)

[12] Beloglazov, A., Buyya, R., "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurrency Computat.*, Pract. Exper. 24, 13971420, 2012. Article (CrossRef Link)

[13] Igo Goiri, Berral, J.L., Fit, J.O., Juli, F., Nou, R., Guitart, J., Gavald, R., Torres, J., "Energy-efficient and multifaceted resource management for profit-driven virtualized data centers," *Future Generation Computer Systems*, 2012. Article (CrossRef Link)
[14] Stillwell, M., Schanzenbach, D., Vivien, F., Casanova, H., "Resource allocation algorithms for virtualized service hosting platforms," *Journal of Parallel and Distributed Computing*, 70(9), pp. 962 – 974, 2010. Article (CrossRef Link)
[15] Aisopos, F., Tserpes, K., Varvarigou, T., "Resource management in software as a service using the knapsack problem model," *International Journal of Production Economics*, 2011. Article (CrossRef Link)
[16] Nguyen Van, H., Dang Tran, F., Menaud, J.M., "Autonomic virtual resource management for service hosting platforms," in *Proc. of Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, CLOUD '09, Washington DC, USA, pp. 1–8, 2009. Article (CrossRef Link)
[17] Lin, W., Qi, D., "Research on resource self-organizing model for cloud computing," in *Proc. of 2010 International Conference on Internet Technology and Applications*, pp. 1–5, 2010. Article (CrossRef Link)
[18] Teng, F., Magoule, s. F., "Resource pricing and equilibrium allocation policy in cloud computing," in *Proc. of 2010 IEEE 10th International Conference on Computer and Information Technology (CIT)*, pp. 195 –202, 2010. Article (CrossRef Link)
[19] Wei, G., V., V.A., Yao, Z., Xiong, N., "A game-theoretic method of fair resource allocation for cloud computing services," *J. Supercomput*, 54, pp. 252–269, 2010. Article (CrossRef Link)
[20] Hassan, M.M., Hossain, M., Sarkar, A., Huh, E.N., "Cooperative game-based distributed resource allocation in horizontal dynamic cloud federation platform," *Information Systems Frontiers*, pp. 1–20, 2012. Article (CrossRef Link)
[21] Berral, J. L., Gavalda, R., & Torres, J., "Adaptive scheduling on power-aware managed data-centers using machine learning," in *Proc. of Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*, IEEE Computer Society, pp. 66-73, 2011. Article (CrossRef Link)
[22] Younge, A. J., Von Laszewski, G., Wang, L., Lopez-Alarcon, S., & Carithers, W., "Efficient resource management for cloud computing environments," in *Proc. of IEEE Green Computing Conference*, pp. 357-364, 2010. Article (CrossRef Link)
[23] Sembiring, K., & Beyer, A., "Dynamic resource allocation for cloud-based media processing," in *Proc. of Proceeding of the 23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, ACM, pp. 49-54, 2013. Article (CrossRef Link)
[24] Nan, X., He, Y., & Guan, L., "Optimal resource allocation for multimedia application providers in multi-site cloud," in *Proc. of 2013 IEEE International Symposium on Circuits and Systems (ISCAS)*,IEEE, pp. 449-452, 2013. Article (CrossRef Link)
[25] Nan, X., He, Y., Guan, L., "Optimal resource allocation for multimedia cloud based on queuing model," in *Proc. of* 2011 *IEEE 13th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1 –6, 2011. Article (CrossRef Link)
[26] Nan, X., He, Y., Guan, L., "Optimal allocation of Virtual Machines for cloud-based multimedia applications," in *Proc. of 2012 IEEE 14th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 175–180, 2012. Article (CrossRef Link)
[27] Nan, X., He, Y., Guan, L., "Optimal resource allocation for multimedia cloud in priority service scheme," in *Proc. of 2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1111 –1114, 2012. Article (CrossRef Link)
[28] Wen, H., Hai-ying, Z., Chuang, L., Yang, Y., "Effective load balancing for cloud-based multimedia system," in *Proc. of 2011 International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT)*, vol. 1, pp. 165–168, 2011. Article (CrossRef Link)
[29] Kou, L.T., Markowsky, G., "Multidimensional bin packing algorithms," *IBM J. Res.*, Dev. *21*, pp. 443–448, 1977. Article (CrossRef Link)

[30] Calyam, P., Patali, R., Berryman, A., Lai, A.M., Ramnath, R., "Utility directed resource allocation in virtual desktop clouds," *Comput. Netw.*, 55, 4112–4130, 2011. Article (CrossRef Link)
[31] Lai, G., Song, H., Lin, X., "A service based light weight desktop virtualization system," in *Proc. of 2010 International Conference on Service Sciences (ICSS)*, pp. 277–282, 2010. Article (CrossRef Link)
[32] Chen, G., He, W, B., Liu, J., Nath, S., and Leonidas, R., Lin, X., Feng, Z., "Energy- aware server provisioning and load dispatching for connection-intensive internet services," in *Proc. of Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, vol. 5, pp. 337–350, 2008
[33] Xiao, Z., Song, W, J., Chen, Qi., "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment," in *Proc. of IEEE Transactions on Parallel and Distributed Systems*, vol. 24, number. 6, pp. 1107–1117, 2013. Article (CrossRef Link)
[34] Bisection Method. http://en.wikipedia.org/wiki/Bisection_method. Access date: 2013
[35] Haskell., Barry, G., "Digital video: an introduction to MPEG-2," Publisher: Kluwer Academic Pub, 1997
[36] Mitchell, Joan, L., "MPEG video compression standard," Publisher: Kluwer Academic Pub, 1996
[37] Hossain, M. S., Hassan, M. M., Qurishi, M. A., and Alghamdi, A., "Resource allocation for service composition in cloud-based video surveillance platform," in *Proc. of IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pp. 408-412, 2012. Article (CrossRef Link)
[38] Ouyang, Z., Xu, L., Ramamurthy, B., "Diverse community: Demand differentiation in P2P live streaming," *Peer-to-Peer Networking and Applications*, 4(1), pp. 23-36, 2011. Article (CrossRef Link)
[39] Zhou, L., Chen, M., Qian, Y., Chen, H., "Fairness Resource Allocation in Blind Wireless Multimedia Communications," *IEEE Transaction on Multimedia*, 15(4), pp. 946-956, 2013. Article (CrossRef Link)
[40] Louvel, M., Plantec, A., Babau, J. P., "Resource management for multimedia applications, distributed in open and heterogeneous home networks," *Journal of Systems Architecture*, 59(3), pp. 121-134, 2013. Article (CrossRef Link)

Mohammad Mehedi Hassan is an Assistant Professor of Information Systems Department in the College of Computer and Information Sciences, King Saud University, Riyadh, Kingdom of Saudi Arabia. He received his Ph.D. degree in Computer Engineering from Kyung Hee University, South Korea in February 2011. He was a Research Professor at Computer Engineering department, Kyung Hee University, South Korea from March, 2011 to October, 2011. He has authored and co-authored more than 50 publications including refereed IEEE/ACM/Springer journals, conference papers, books, and book chapters. He has served as, chair, and Technical Program Committee member in numerous international conferences/workshops like IEEE HPCC, IEEE ICME, ACM Multimedia, ICA3PP, IEEE ICC, TPMC, IDCS, etc. His research interests include Cloud collaboration, multimedia Cloud, sensor-Cloud, mobile Cloud, Thin-Client, Grid computing, IPTV, virtual network, sensor network, and publish/subscribe system.

Biao Song received a B.S. degree from the Department of Computer Sciences, Hebei Normal University, in 2008. He received his Ph.D. degree in Computer Engineering from Kyung Hee University, South Korea in August 2012. Currently he is an Assistant Professor of Information Systems Department in the College of Computer and Information Sciences, King Saud University, Riyadh, Kingdom of Saudi Arabia. His research interests include task co-allocation, dynamic collaboration in cloud computing, remote display protocols in thin client environments, and IPTV service delivery over virtual networks.

M. Shamim Hossain is an Associate Professor of CCIS, at the King Saud University, Riyadh, KSA. He received his Ph.D. in Electrical and Computer Engineering from the University of Ottawa, Canada. His research interests include Service oriented computing, Collaborative Cloud media, Service configuration, and biologically inspired approach for multimedia and distributed system. He served as guest editor for IEEE Transactions on Information Technology in Biomedicine, and Springer Multimedia tools and Applications. He is a Senior Member of IEEE and a member of ACM.

Atif Alamri is an Assistant Professor and currently the Chairman of Information Systems Department, at the College of Computer and Information Sciences, King Saud University. Riyadh, Saudi Arabia. His research interest includes multimedia assisted health systems, ambient ntelligence, and service-oriented architecture. Mr. Alamri was a Guest Associate Editor of the IEEE RANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, a Cochair of the first IEEE International Workshop on Multimedia Services and Technologies for E-health, a Technical Program Cochair of the 10th IEEE International Symposium on Haptic Audio Visual Environments and Games, and serves as a Program Committee Member of many conferences in multimedia, virtual environments, and medical applications.

Mohammed Alnuem is an Assistant Professor of Information Systems Department, at the College of Computer and Information Sciences, King Saud University. Riyadh, Saudi Arabia. He is currently the Vice Dean for Quality and Development at e-Transactions and Communications Deanship, King Saud University. He received his PhD degree in Computer Sciecne from Bradford University, UK in 2009. He has authored and co-authored many publications including refereed IEEE/ACM/Springer journals, conference papers, books, and book chapters. His research interest includes sensor network, cloud computing, mutimedia, RFID and security.

Muhammad Mostafa Monowar is currently working as an Assistant Professor at Department of Information Technology, Faculty of Computing and Information Technology in King AbdulAziz University, Kingdom of Saudi Arabia. He received his Ph.D. degree in Computer Engineering in 2011 from Kyung Hee University, South Korea. He received B.Sc. degree in Computer Science and Information Technology from Islamic University of Technology (IUT), Bangladesh in 2003. His research interest includes Wireless networks, especially ad hoc, sensor and mesh networks, including routing protocols, MAC mechanisms, IP and transport layer issues, cross-layer design and QoS provisioning. . He has served as a program committee member in several international conferences/workshops like IADIS, DNC, IDCS etc. He is currently serving as an Associate Editor of IJIDCS, Guest Editor of some special issues of IJCSE.

M. Anwar Hossain received the B.Sc. Engg. degree in computer science and engineering from Khulna University, Khulna, Bangladesh, in 1995, and the M.C.S. degree in computer science and the Ph.D. degree in electrical and computer engineering both from the University of Ottawa, Ottawa, ON, Canada, in 2005 and 2010, respectively. He is currently an Assistant Professor in the Department of Software Engineering, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia. At University of Ottawa, he was associated with the Multimedia Communications Research Laboratory (MCRLab), School of Information Technology and Engineering. His research interests include ambient intelligence, multisensor surveillance, pervasive healthcare, and multimedia cloud computing. He has authored or coauthored more than 50 publications including refereed journals, conference papers, and book chapters.