ISSN 1392–124X (print), ISSN 2335–884X (online) INFORMATION TECHNOLOGY AND CONTROL, 2015, T. 44, Nr. 1

An Efficient Scheduling Strategy for Batch Processing Applications in Mobile Cloud: Model and Algorithm

Chunlin Li, LaYuan Li

Department of Computer Science, Wuhan University of Technology,
Wuhan 430063, P.R.China
e-mail: chunlin74@aliyun.com, jwtu@public.wh.hb.cn

crossref http://dx.doi.org/10.5755/j01.itc.44.1.5319

Abstract. Mobiles enter cloud computing domain by trying to access the shared pool of computing resources provided by the cloud on demand. Mobile cloud computing brings new types of services and facilities for mobile users to take full advantage of cloud computing. The paper considers batch processing applications for mobile cloud computing environment. The mobile device's user requirements arrive in batches into the mobile cloud systems. For example, mobile device's users submit batch jobs (e.g., financial analytics, scientific simulations) to mobile cloud system for fast processing. The paper proposes a multistage scheduling for batch processing applications in mobile cloud. The multistage scheduling optimization is involved with mobile cloud provider's optimization, mobile device application's optimization and mobile device job's optimization, respectively. Multimedia search as an example in mobile cloud environment is presented, and the proposed multistage scheduling method is applied to mobile cloud environment. In the simulations, our proposed mobile cloud multistage scheduling algorithms are compared with two related works. Our algorithm combines the perspectives of mobile cloud providers and mobile device users; it outperforms better than other related works.

Keywords: mobile cloud computing; batch processing; multistage optimization model.

1. Introduction

As mobile devices become increasingly powerful, mobile devices extends beyond traditional telecommunications and moves to cloud computing environment. However, low bandwidth, intermittent network connectivity and scarcity of computing resources and energy are still key issues in applying mobiles in complex and data intensive applications. Mobile cloud architecture can facilitate enormous amounts of data storage and high computational capabilities by means of the Cloud [1]. The purpose of mobile cloud computing is to balance the application distribution between the mobile device and the cloud, in order to achieve faster interactions, battery savings and better resource utilization. With this, mobile devices evolve from being mere intermediaries between the cloud and the end user into true intermediaries of cloud computing.

Given the nature of cloud applications, users do not need to have the highest resource devices, as complex computing operations would be run within the cloud. This lessens the cost of mobile computing to the client and allows even low-entry types of devices to take advantage of the cloud capabilities. Mobile cloud can facilitate the use of mobile devices to collect data, manipulate them and interact with scientific workflows running in the Cloud. By deploying data-intensive computation and data storage to the Cloud, the mobile cloud can release mobiles from heavy computational loads, thereby reducing mobile energy consumption, while using the cloud to increase processing power and storage capacity.

There are some works dealing with mobile cloud. In [2], Kaewpuang et al. proposed a framework for resource allocation to the mobile applications, and revenue management and cooperation formation among service providers. They formulate and solve optimization models to obtain the optimal number of application instances that can be supported to maximize the revenue of the service providers while meeting the resource requirements of the mobile applications. In [3], Wu et al. studied the tradeoff between shortening execution time and extending battery life of mobile devices in mobile cloud. A novel adaptive offloading scheme is proposed and analyzed based on the tradeoff analysis. In [4], Yamauchi et al. proposed a distributed parallel scheduling methodology for mobile cloud and developed a simulator to analyze these characteristics and the bottleneck of mobile cloud. In [5], Lin et al. propose an optimal control policy in a mobile cloud computing system based on stochastic data. They define the expected "performance sum" as the objective function, which essentially captures a desirable trade-off between performance and power consumption of the mobile device. Abolfazli et al. [6] propose a market-oriented architecture based on SOA (service-oriented architecture) to stimulate publishing, discovering, and hosting services on nearby mobiles. In [7], a framework from modeling to design, and to implementation is proposed to build a service selection system in mobile cloud. A Markov chain model is used for performance measures calculation. In [8], Park and Lee make groups of mobile devices by measuring the behavior of mobile devices and calculating the entropy in mobile cloud. In [9], Nishio et al. propose an architecture and mathematical framework for heterogeneous resource sharing in mobile cloud. They formulate optimization problems for maximizing the sum of the utility functions and solve them via convex optimization approaches.

Shiraz et al. [10] study virtual machine deployment for application outsourcing in mobile cloud. This paper analyzes the impact of VM (Virtual Machine) deployment and management on the execution time of application. In [11], Balakrishnan and Tham attempt to apply DVFS (Dynamic voltage and frequency scaling) in mapping as well as scheduling stages by combining both the task-resource and resource-frequency assignments in mobile cloud. In [12], Mohammad et al. propose a cooperative game-theoretic solution for the benefit of the cloud providers in horizontal dynamic cloud federation. They study two utility maximizing cooperative resource allocation games. In [13], Hung et al. present a novel architecture, taking advantage of collaboration of thin and thick clients in cloud computing. The paper aims at optimizing data distribution and utilizing cloud resources so that QoS (Quality of Service) requirements can be met. They also propose an algorithm to select an optimal resource allocation strategy to satisfy various Service Level Agreements. In [14], Yang et al. study how to optimize the computation partitioning of a data stream application between mobile and cloud to achieve maximum speed/throughput in processing the streaming data in mobile cloud. Li et al. [15] propose Armada, an efficient range query processing scheme to support delay bounded single-attribute and multiple-attribute range queries. Sanaei et al. [16] propose a Service-based arbitrated multi-tier infrastructure for mobile cloud computing.

In [17], Sindia et al. explore how cloud computing techniques can be used on mobile devices. Two ways are proposed to deploy mobile cloud computing in an efficient manner: a customizable job scheduler; and a mobile friendly MapReduce framework. In [18], Niyato et al. model the resource allocation process of a mobile cloud computing system as an auction mechanism with premium and discount factors. In [19], Park et al. propose a resource allocation technique which offers reliable resource allocation considering the availability of mobile resources and movement reliability of mobile resources in mobile cloud. Reference [20] proposed phased scheduling for resource-constrained mobile devices in mobile cloud computing. Reference [21] presents optimal resource provisioning for cloud computing environment. Andziulis et al. [22] study robust intelligent construction procedure for job-shop scheduling.

From the above review of related literature on mobile cloud, most researches of the mobile cloud scheduling do not consider how to fulfill both mobile users' expectations and mobile cloud providers' optimization objectives. The methods and contributions of this paper are different from the above related works. Our contributions are as follows.

1) The formulation of mobile cloud multistage scheduling strategy for batch processing applications combines the perspectives of mobile cloud providers and mobile device users.

2) The maximization of the Lagrangian of mobile cloud multistage scheduling optimization in mobile cloud can be processed in parallel. In order to achieve a distributed solution, the multistage scheduling optimization is involved with mobile cloud provider's optimization, mobile device application's optimization and mobile device job's optimization, respectively.

3) The paper adopts a distributed mobile cloud multistage scheduling algorithm among mobile device's batch applications, mobile cloud providers and mobile device users in mobile cloud.

In the paper, multimedia search as an example in mobile cloud environment is presented, and the proposed multistage scheduling method is applied to mobile cloud environment. The experiments aim at comparing our algorithm (*MCMSA*) with other two related works. The rest of the paper is structured as follows. Section 2 discusses system model of batch processing applications for mobile cloud computing. Section 3 presents mobile cloud multistage scheduling algorithm. In Section 4, the experiments are conducted and discussed. Section 5 gives an application example. Section 6 gives the conclusions to the paper.

2. Efficient Scheduling Strategy for Batch Processing Applications in Mobile Cloud

2.1. System Model Description

The mobile cloud system proposed in Fig. 1 includes mobile device users, cloud datacenter and mobile cloud proxy. For achieving efficient scheduling optimization for batch processing applications in mobile cloud, different multistage scheduling strategies are deployed at three levels: mobile cloud job scheduling, mobile cloud batch applications' scheduling and mobile cloud system's scheduling. At the top, the mobile cloud system scheduling controls the gross provisioning of VMs to the mobile cloud batch applications. At the next level down, the mobile cloud batch applications scheduling is responsible for the

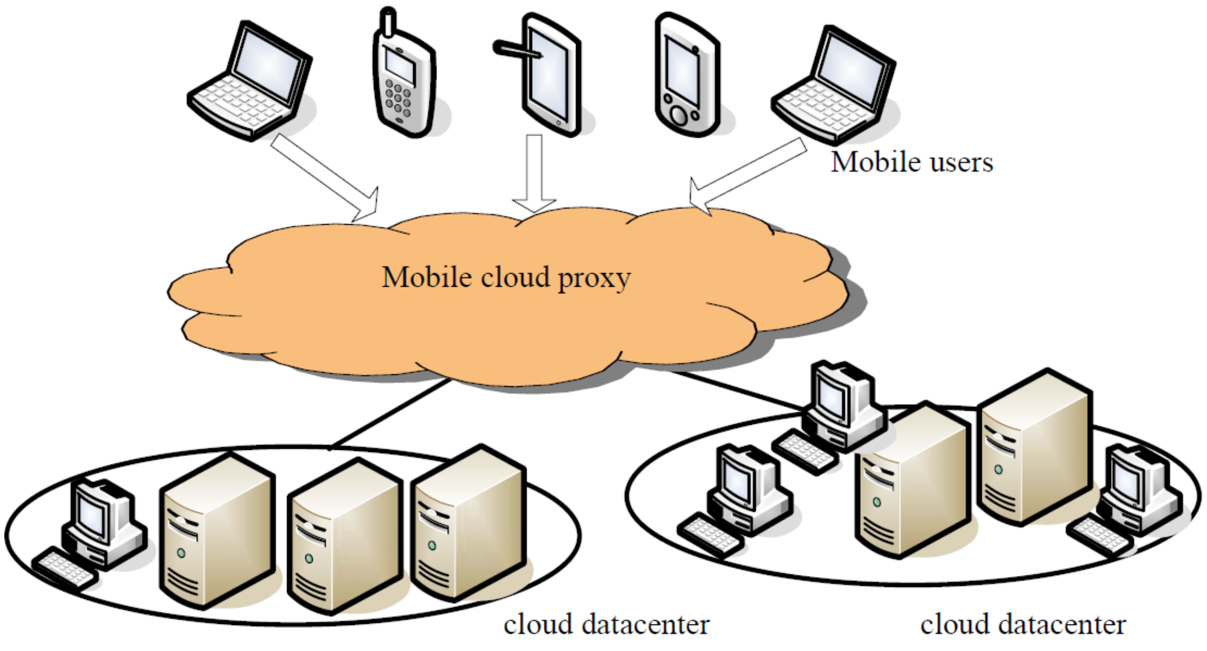

Figure 1. Mobile cloud environment

deployments of all mobile device applications that exploit the mobile cloud resources. At the lowest level, the mobile device's job scheduling adjusts the mobile cloud resource usages to optimize the utility of single mobile device application. Mobile device's batch applications' scheduling take more time to decide which of their control actions will maximize mobile device's batch applications' utility. Mobile cloud job scheduling aims at maximizing the utility of mobile cloud job. The multistage scheduling process in mobile cloud is shown in Fig. 2. The mobile cloud system scheduling performs a system wide allocation of mobile cloud resources. After initialization, the scheduling of mobile device's batch applications and mobile device's job scheduling in the system would be able to take finer control. The mobile cloud system scheduling chooses mobile cloud resource allocation for the mobile device's application that maximizes mobile cloud system utility.

The operations of mobile device's job level scheduling, mobile device's batch applications' scheduling and mobile cloud system's scheduling are coordinated with each other. Efficient multistage scheduling optimization for batch processing applications uses composite utility functions to measure system performance at multistage. Mobile cloud scheduling is deployed at the different stages; the implementation of multistage scheduling optimization leads to the decomposition of the system. The mobile device's batch applications' scheduling works independently to acquire VMs from mobile cloud providers. The mobile device's batch applications' scheduling acquires VMs in order to maintain batch applications' utility.

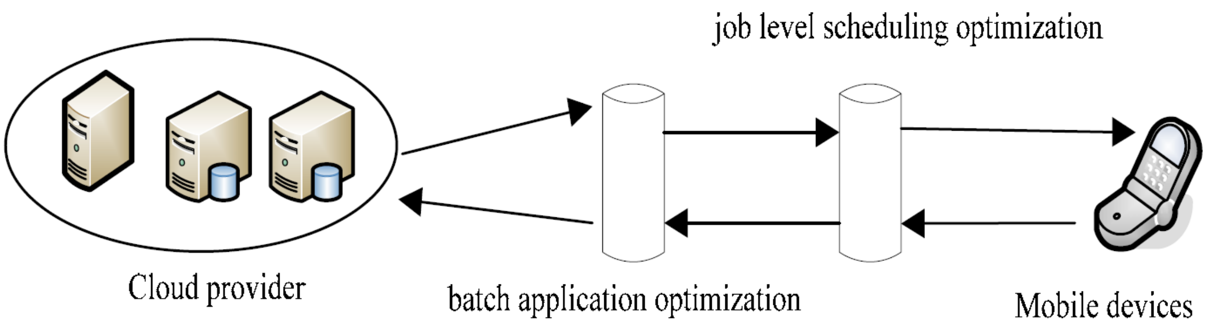
job level scheduling optimization
Cloud provider
batch application optimization
Mobile devices

Figure 2. Multistage scheduling process in mobile cloud

2.2. Problem Formulation

Let v_i^j denote the VM for mobile device's application i from the mobile cloud provider j. The deadline given by the mobile device's batch application k is denoted by T_k. The maximum capacity of mobile cloud provider j is denoted by C_j. The time taken by the mobile device's application i to complete the nth job is denoted by t_i^n. t_k^i refers to the time taken by the i application in the mobile device's batch application k, r_i^j refers to the payments of the mobile device's application i to mobile cloud provider j, E_k refers to the budget of mobile device's batch application k, T_i refers to the deadline given by mobile device's application i. The energy dissipation used by the jth mobile cloud provider is denoted by en_j . The limit of energy consumption of mobile cloud provider j is denoted by D_j. The budget of mobile device's application i is denoted by B_i. The payment of the nth job of mobile device's application i is denoted by s_i^n. The computation task of the ith mobile device's application's nth job is denoted by q_i^n .

In multistage scheduling model of mobile cloud, the objective of mobile device's batch applications optimization is to provision VMs for batch applications such that the mobile cloud utility $U_{Mobilecloud}$ is maximized subject to the resource constraints of mobile cloud datacentre and the requirements of mobile device's batch applications, respectively. The problem of mobile cloud multistage scheduling optimization is formulated as follows:

$$\begin{aligned}&Max\ U_{Mobilecloud}\\&s.t E_k \geq \sum_{i=1}^{I} r_i^j,\\&en_j \leq D_j,\\&C_j \geq \sum_i v_i^j,\\&T_k \geq \sum_{i=1}^{I} t_k^i.\end{aligned} \tag{2.1}$$

Mobile cloud system utility is the sum of the mobile device's batch application's utility and mobile cloud provider' utility. It aims to jointly optimize the benefit of mobile device's application and mobile cloud provider.

$$U_{Mobilecloud} = \sum_{k=1}^{K}\left(\left(T_k - \sum_{i=1}^{I} t_k^i\right) + \left(E_k - \sum_{i=1}^{I} r_i^j\right)\right) + \sum_{i=1}^{N}\left(r_i^j \log v_i^j\right) - en_j. \tag{2.2}$$

In Formula (2.2), N is the sum number of mobile device's applications. K denotes the sum number of the mobile application groups. I denotes total number of mobile device's applications. j denotes certain mobile cloud provider. i denotes certain mobile device's application.

$\left(T_k - \sum_{i=1}^{I} t_k^i\right) + \left(E_k - \sum_{i=1}^{I} r_i^j\right)$ means the mobile device's batch applications' saving time and cost surplus, when completing the mobile device's applications. It is the utility of mobile device's batch application k. $\sum_{i=1}^{N} r_i^j \log v_i^j - en_j$ presents the benefit of mobile cloud provider.

The constraint of Formula (2.2) implies that the aggregate VMs can not exceed the total number of VMs of mobile cloud provider j. Other constraints are related with mobile cloud applications. The objective of mobile cloud applications is to complete a sequence of applications within specified deadline, T_k, while the total payment cannot exceed the budget E_k, $\sum_{i=1}^{I} r_i^j$ are the payments of the mobile device's batch applications to the mobile cloud provider j for provisioned VMs.

Let us consider the Lagrangian form of mobile cloud multistage scheduling optimization problem:

$$L = U_{Mobilecloud} + \lambda\left(C_j - \sum_i v_i^j\right) + \beta\left(T_k - \sum_{i=1}^{I} t_k^i\right) + \mu\left(E_k \sum_{i=1}^{I} r_i^j\right) + \sigma\left(D_j - en_j\right) \tag{2.3}$$

where λ_i , β , μ , σ are the Lagrangian multipliers. Solving Formula (2.2) requires the cooperation of mobile cloud applications, but it is not applicable in mobile cloud environment. Since the Lagrangian function is separable, the maximization of the Lagrangian can be processed in parallel by mobile cloud applications and mobile cloud providers. The mobile cloud batch application optimization problem can be converted into two sub-optimization problems, which are respectively achieved by mobile cloud batch applications and mobile cloud providers as follows:

$$\begin{aligned}&Max\ U_{Cloudprovider} = \sum_{i=1}^{N}\left(r_i^j \log v_i^j\right) - en_j\\&s.t C_j \geq \sum_i v_i^j , en_j \leq D_j.\end{aligned} \tag{2.4}$$

$$\begin{aligned}&Max\ U_{Mobilebatchapp} = \sum_{k=1}^{K}\left(\left(T_k - \sum_{i=1}^{I} t_k^i\right) + \left(E_k - \sum_{i=1}^{I} r_i^j\right)\right)\\&s.t T_k \geq \sum_{i=1}^{I} t_k^i , E_k \geq \sum_{i=1}^{I} r_i^j .\end{aligned} \tag{2.5}$$

Formula (2.4) corresponds to the behavior of the mobile cloud provider. Different mobile cloud providers compete for provisioning the VMs for mobile cloud batch applications and maximizing the revenue. In Formula (2.4), N denotes the total number of mobile cloud providers. $\sum_{i=1}^{N}\left(r_i^j \log v_i^j\right)$ presents the revenue obtained by mobile cloud provider j from mobile device's application i. We chose the log function because the benefit increases quickly from zero as the allocated virtual machines increase from zero and then increases slowly. To provision VMs for mobile cloud batch applications, the mobile cloud provider has to pay for the energy cost. The mobile cloud provider aims to maximize the utility function without exceeding maximal energy constraints. Formula (2.5) corresponds to the behavior of the mobile cloud batch applications. The mobile batch applications compute the optimal payment to mobile cloud providers under the constraints to maximize mobile device's batch applications' satisfaction. The ith mobile device's application pays r_i^j to the mobile cloud provider j for obtaining virtual machines. $E_k - \sum_{i=1}^{I} r_i^j$ represents the surpluses of mobile device's batch applications. $\left(T_k - \sum_{i=1}^{I} t_k^i\right)$ represents the saving time, which is calculated by the deadline minus the actual execution time.

In job-level optimization problem of mobile cloud multistage scheduling model, a mobile device's application needs to complete a sequence of jobs within the deadline, T_i, while minimizing the cost occurred and processing time. Each job of mobile device's application i submits s_i^n for the VM. Mobile device's jobs compete for the resources of mobile cloud application i. The resources allocated to mobile device's jobs depend on the relative payments sent by all jobs. The nth mobile device's jobs receive mobile cloud resources proportional to its payment. Given the deadline T_i for mobile device's application i to complete all jobs, the job-level optimization scheduling in mobile cloud can be formulated as:

$$Max\ U_{mobiledevicejob} = \left\{\left(B_i - \sum_n s_i^n\right) + \sigma\left(T_i - \sum_n t_i^n\right)\right\}$$
$$s.t T_i \geq \sum_n t_i^n. \qquad (2.6)$$

In Formula (2.6), σ is the relative importance of costs and times to complete mobile device jobs, mobile cloud application with larger value of σ would indicate a greater preference to reduce its completion time. When $\sigma = 1$, meaning that costs and times are equally important. In Formula (2.6), we use absolute value of money and time, and the weights of the factors are equal.

2.3. Solutions for Scheduling Strategy for Batch Processing Applications in Mobile Cloud

For multistage scheduling for batch processing applications in mobile cloud, the mobile cloud system scheduling controls the gross provisioning of VMs to the mobile device's batch applications. Mobile cloud provider's optimization aims at computing the optimal VM v_i^{j*} for mobile device's batch applications while maximizing the benefit function of mobile cloud provider without exceeding the total number of VMs and upper payment of energy consumption. The VMs allocated to mobile device's batch applications are constrained by the total of capacity of mobile cloud providers. Total allocated VMs do not exceed the total capacity C_j. For the mobile cloud provider's optimization problem, mobile cloud providers compute optimal VMs to maximize the benefit function and minimize the payment for providing VMs to mobile device's batch applications. The profits of mobile cloud provider are affected by the payments of mobile device's batch applications and energy payment of provisioning VMs. So the revenue of mobile cloud provider increases when the VMs leased to the mobile device's batch applications increase and the payments received from mobile devices increase, also the payment for energy consumption decreases. The sum $\sum_{i=1}^{N}\left(r_i^j \log v_i^j\right)$ presents the revenue obtained by mobile cloud provider j from mobile device's batch applications. The objective of mobile cloud provider is to maximize the revenue and minimize energy consumption en_j.

Mobile device's batch application adaptively submits the demand of VM based on the current conditions of mobile cloud provider, while the mobile cloud provider adaptively allocates VMs required by the mobile device's batch applications. The interaction between mobile device's batch applications and mobile cloud provider is controlled through the use of the variables p_j, p_j denotes VM prices provided by the mobile cloud providers, which is used in mobile cloud batch application optimization problem.

The energy consumption rate of mobile cloud provider for hosting VMs is denoted as e_j. The energy consumption of mobile cloud provider j to provision virtual machine for mobile device's application i denoted as ec_i^j can be written as follows:

$$ec_i^j = e_j * v_i^j. \qquad (2.7)$$

Let px_j denote electricity price. The energy consumption cost of mobile cloud provider for hosting VMs is expressed as follows:

$$en_j = px_j * \sum_{i=1}^{N} ec_i^j. \qquad (2.8)$$

The mobile cloud provider's optimization problem is reformulated as

$$Max \sum\left(r_i^j \log v_i^j\right) - px_j \sum_{i=1}^{N} e_j * v_i^j. \qquad (2.9)$$

The Lagrangian function for $U_{cloudprovider}(v_i^j)$ in Formula (2.4) is

$$L(v_i^j) = \sum\left(r_i^j \log v_i^j\right) - px_j \sum_{i=1}^{N} e_j * v_i^j + \lambda\left(C_j - \sum_i v_i^j\right) + \eta\left(D_j - px_j \sum_{i=1}^{N} e_j * v_i^j\right) \quad (2.10)$$

where λ and η are the Lagrangian constants. From Karush-Kuhn-Tucker Theorem, the optimal solution can be gotten, given $\partial L(v_i^j) / \partial v_i^j = 0$ for $\lambda > 0$. We take derivative with respect to v_i^j and get (2.11) as follows:

$$\partial L(v_i^j) / \partial v_i^j = \frac{r_i^j}{v_i^j} - (1 + \lambda + \eta) px_j e_j \qquad (2.11)$$

Let $\partial L(v_i^j) / \partial v_i^j = 0$, we can get v_i^j as follows:

$$v_i^j = \frac{r_i^j}{(1+\lambda+\eta)px_je_j} \tag{2.12}$$

Using this result in the constraint equation of Formula (2.4), let $\omega = 1 + \eta + \lambda$, we can get the result as follows:

$$D_j = \frac{1}{\omega}\sum r_i^j, \quad \omega = \frac{\sum r_i^j}{D_j}$$

From Karush-Kuhn-Tucker Theorem, we can get $v_i^{j^*}$ as follows:

$$v_i^{j^*} = \frac{r_i^j D_j}{px_je_j\sum r_i^j} \tag{2.13}$$

In (2.13), $v_i^{j^*}$ means that the mobile cloud provider j computes optimal units of virtual machines for mobile device's application i while maximizing its benefit.

The mobile device's application i is the consumer of mobile cloud provider, which provision VMs for mobile device's application. The mobile device's application i submits payment r_i^j to the mobile cloud provider j for VM. Let r_i^j be the payment of the ith mobile device's application. N mobile device's applications compete for the VMs. Mobile cloud provider's VMs are allocated using a market mechanism, where the divisions depend on the relative payments sent by the mobile device's batch applications.

Let's consider the interactions of mobile device's application i and mobile cloud provider in mobile cloud. The benefit function for mobile device's application i depends on the units of VM denoted by v_i^j. For the mobile device's batch application optimization problem, the mobile device's application i calculates the unique optimal payment to mobile cloud provider under the constraints to maximize the mobile device's batch application's benefit. The payment accrued to buy or lease VMs cannot exceed the budget of mobile device's batch application E_k.

The mobile device's batch applications give the unique optimal payment to mobile cloud providers under the constraints of the deadline and budget to maximize the set of mobile device's batch applications' benefits.

The time taken by the mobile device's batch application to complete the ith application is

$$t_k^i = \frac{p_j}{c_jr_i^j}. \tag{2.14}$$

The mobile cloud batch application optimization problem can be rewritten as follows:

$$Max\left(T_k - K\sum_i \frac{p_j}{c_jr_i^j}\right) + \left(E_k - \sum_j r_i^j\right)$$

$$s.t T_k \geq \sum_{i=1}^I t_k^i, E_k \geq \sum_{i=1}^I r_i^j. \tag{2.15}$$

In (2.15), k denotes the number of mobile cloud batch application, j denotes certain mobile cloud provider. $E_k \geq \sum_{i=1}^I r_i^j$ means the payments of the ith mobile cloud application in the mobile device's batch application k to the mobile cloud provider j can not exceed the budget E_k.

Let the pricing policy, $p = (p_1, p_2, \dots, p_J)$, denote the set of VM prices of all mobile cloud providers. The mobile device's application i receives the VMs according to its payment relative to the sum of the mobile cloud provider's revenue.

The Lagrangian for the problem $U_{Mobilebatchapp}$ in Formula (2.5) is $L(r_i^j)$

$$L(r_i^j) = \left(E_k - \sum_j r_i^j\right) + \left(T_k - K\sum_j \frac{p_j}{c_jr_i^j}\right) +$$

$$\beta\left(E_k - \sum_j r_i^j\right) + \eta\left(T_k - K\sum_j \frac{p_j}{c_jr_i^j}\right) \tag{2.16}$$

where β and η are the Lagrangian constants. From Karush-Kuhn-Tucker Theorem, the optimal solution can be gotten, if $\partial L / \partial r_i^j = 0$ for $\beta > 0$. We take derivative with respect to r_i^j and get (2.17) as follows:

$$\partial L(r_i^j) / \partial r_i^j = -1 + K\frac{p_j}{c_j(r_i^j)^2} - \beta +$$

$$\eta K\frac{p_j}{c_j(r_i^j)^2} \tag{2.17}$$

Let $\partial L / \partial r_i^j = 0$, we get r_i^j denoted in (2.18) as follows:

$$r_i^j = \left(\frac{(k\eta+k)p_j}{(1+\beta)c_j}\right)^{1/2}. \tag{2.18}$$

Using this result in the constraint equation in (2.15), let $\theta = \frac{(k\eta+k)}{(1+\beta)}$, we can get $(\theta)^{-1/2}$

$$(\theta)^{-1/2} = \frac{T_k}{\sum_{j=1}^J \left(\frac{p_j}{c_j}\right)^{1/2}}. \tag{2.19}$$

We use the result of (2.19) and apply it to (2.18) to obtain $r_i^{j^*}$ as follows:

$$r_i^{j^*} = \left(\frac{p_j}{c_j}\right)^{1/2} \frac{\sum_{j=1}^J \left(\frac{p_j}{c_j}\right)^{1/2}}{T_k}. \tag{2.20}$$

The mobile cloud application i pay $r_i^{j^*}$ to mobile cloud provider j for virtual machines.

In mobile cloud multistage scheduling model, mobile device's job level scheduling optimization in mobile cloud is conducted by mobile device's application; the mobile device's application calculates the payment to mobile cloud provider under the deadline to satisfy the mobile cloud application's requirements. $B_i - \sum_n s_i^n$ is the surplus of all jobs of mobile device's application. $\sum_n t_i^n$ represents the execution time for processing all mobile device application's jobs. The objective of job level scheduling optimization is to minimize the cost of mobile device's applications and complete all jobs as soon as possible. Under the constraint of the deadline, mobile device application i wants to complete all jobs. q_i^n is the computation task of ith mobile device application's nth job. The execution time taken by the ith mobile device application to complete the nth job is:

$$t_i^n = \frac{q_i^n}{v_i^j s_i^n}. \qquad (2.21)$$

The mobile device's job level scheduling optimization is reformulated as

$$Max\left\{\left(B_i - \sum_n s_i^n\right) + \left(T_i - \sum_{n=1}^{N} \frac{q_i^n}{v_i^j s_i^n}\right)\right\}. (2.22)$$

The Lagrangian for $U_{mobiledevicejob}$ in (2.6) is $L(s_i^n)$.

$$L(s_i^n) = \left(B_i - \sum_n s_i^n\right) + \left(T_i - \sum_{n=1}^{N} \frac{q_i^n}{v_i^j s_i^n}\right) + \lambda\left(T_i - \sum_{n=1}^{N} t_i^n\right) \qquad (2.23)$$

where λ is the Lagrangian constant.

Using this result in the constraint equation in (2.6), Let $\theta = 1 + \lambda$, θ can be obtained in (2.24)

$$(\theta)^{-1/2} = \frac{T_i}{\sum_{n=1}^{N} \left(\frac{q_i^n}{v_i^j}\right)^{1/2}}. \qquad (2.24)$$

We use the result of (2.24) and apply it to (2.23) to obtain $s_i^{n^*}$

$$s_i^{n^*} = \left(\frac{q_i^n}{v_i^j}\right)^{1/2} \frac{\sum_{n=1}^{N}\left(\frac{q_i^n}{v_i^j}\right)^{1/2}}{T_i}. \qquad (2.25)$$

In (2.25), the nth job of mobile device's application i pay $s_i^{n^*}$ to the mobile cloud provider j.

3. Mobile Cloud Multistage Scheduling Algorithm

The multistage scheduling algorithms in mobile cloud are involved with mobile cloud provider's optimization, mobile device application's optimization and mobile device job's optimization, respectively. The proposed algorithm that achieves multistage scheduling for data-Intensive batch applications can be described as follows.

Algorithm 1. Mobile Cloud Multistage Scheduling Algorithm (MCMSA)

```
Routine mobile device application_Optimization($p_j$)
If    $E_k \ge \sum_{i=1}^{I} r_i^j$
        Then $r_i^{j^*} \leftarrow Max\ U_{mobiledevicejob}(r_i^j)$; //compute $r_i^j$ according to Formula (2.20)
Return $r_i^{j^*}$;
Routine mobile device job Optimization($v_i^j$)
    If  $B_i \ge \sum_{n=1}^{N} s_i^n$
        $s_i^{n^*} \leftarrow Max\ U_{MDJ}(s_i^n)$; // compute $s_i^{n^*}$ according to Formula (2.25)
Return $s_i^{n^*}$;
```

```
Routine mobile cloud provider_Optimization ($r_i^{j^{(n)}}$)
    $v_i^{j^*} \leftarrow Max\ \{U_{Cloudprovider}\}$; //compute $v_i^{j^*}$ according to Formula (2.13)
    If $C_j \ge \sum_i v_i^j$ ;
        Then $p_j^{(n+1)} \leftarrow \max\left\{\varepsilon, p_j^{(n)} + \eta\left(\sum_i v_i^j - C_j\right)\right\}$
    // where $\eta > 0$ is a small step size parameter, $n$ is iteration step. Let $\varepsilon > 0$ be a sufficiently small constant preventing prices to
    approach zero. It is consistent with the law of supply and demand: if the demand for VM exceeds the mobile cloud provider's
    supply $C_j$, then the price $p_j^{(n+1)}$ is raised; otherwise, the VM's price is reduced;
Return $p_j^{(n+1)}$;
```

4. Experiments

In this section, we compare the proposed mobile cloud multistage scheduling algorithm (*MCMSA*) with other related works. A mobile cloud environment with a 2-dimensional area of 500m*500m is used to compare and analyze three algorithms. Mobile cloud proxy residing in WLANs acts as the interface point between the mobile devices. All Wi-Fi interfaces operate at a rate of 11Mb/s. All Ethernet interfaces operate at a rate of 10Gb/s. Jobs arrive at each cloud node s_i, $i = 1, 2, \dots, n$, according to a Poisson process with rate α. The energy cost can be expressed in the dollar that can be defined as unit energy processing cost. Mobile device users submit their jobs with varying deadlines. The deadlines of mobile device user are chosen from 100ms to 400ms. The budgets of mobile device users are set from 100 to 1500 dollars. Each experiment is repeated 6 times and 95% confidence intervals are obtained. Simulation parameters are listed in Table 2.

Table 2. Simulation Parameters

Simulation Parameter	**Value**
Total number of mobile device users	40
Total number of cloud providers	12
Mobility model	Random-walking mobility
Average speed of mobile device	5m/s
Initial price of VM	[10, 500]
Deadline	[100, 400]
Expense budget	[100, 1500]
electrical energy	[0.1, 1.0]
Bandwidth	[100, 1000]
Computing power	[100, 1000]
RAM	[100, 2000]
Energy price	[1, 100]
Job arrival rate	[0.1, 0.6]

The simulations are conducted to compare our mobile cloud multistage scheduling Algorithm (*MCMSA*) with cooperative resource allocation algorithm in mobile cloud computing [2], which is named as *CRAA* and optimal collaboration of thin–thick clients and resource allocation algorithm in mobile cloud computing [13], which is named as *OCRA*. In [2], Kaewpuang et al. propose a framework for resource allocation to the mobile applications, and revenue management and cooperation formation among service providers in mobile cloud. They formulate and solve optimization models to obtain the optimal number of application instances that can be supported to maximize the revenue of the service providers while meeting the resource requirements of the mobile applications. In [13], Hung et al. present a novel architecture that enhances mobile client's capabilities with computing resources from mobile clouds. Hung et al. [13] focus on optimizing the data distribution from cloud networks to mobile client and utilizing computing resources so that QoS requirements can be fulfilled. They proposed an algorithm that can select the best resource allocation strategy in order to satisfy Service Level Agreements.

The following simulation metrics are adopted: resource allocation efficiency, execution success ratio and revenue. Resource allocation efficiency is the ratio of the consumed cloud resources to the total cloud resources available as a percentage. Execution success ratio is the percentage of jobs executed successfully before their deadline. We compare *MCMSA* algorithm with *CRAA* and *OCRA* by varying load factor, deadline and network latency to study how they affect the performance of these algorithms.

Figures 3 to 5 are to study allocation efficiency, execution success ratio, and revenue under different load factor (α), respectively. Fig. 3 shows that when load factor increases ($\alpha = 0.5$), resource allocation efficiency of *MCMSA* is as much as 12% less than that with $\alpha = 0.1$. The resource allocation efficiency is larger when the load factor is higher. When the load factor is 0.5 ($\alpha = 0.5$), resource allocation efficiency of *CRAA* is 22% more than *MCMSA*. Compared with *OCRA*, the resource allocation efficiency of *MCMSA* and *CRAA* sharply decreases than *CRAA* when the load factor increases. When the load factor is 0.7 ($\alpha = 0.7$), the allocation efficiency of *CRAA* decreases to 55%, the energy allocation efficiency of *MCMSA* decreases to 75%. When the load factor increases, fewer mobile cloud users can be admitted into the mobile cloud system due to the increase of system burden, so, resource allocation efficiency decreases. Considering the execution success ratio, from the results in Fig. 4, when load factor is 0.7 ($\alpha = 0.7$), the execution success ratio of *CRAA* is 22% less than that using *MCMSA*. When load factor increases, system load increases as well; some mobile device user's requirements can't be processed on time, this leads to low execution success ratio. When load factor increases, execution success ratio of *CRAA* deteriorates quickly. *OCRA* performs better than *CRAA* and *MCMSA*. *CRAA* resource allocation algorithm does not consider the optimization of both mobile cloud providers and mobile device users; it wants to optimize the revenue of the mobile cloud providers. Fig. 5 shows the effect of the load factor on the revenue. The revenue increases as the load factor increases. When load factor is 0.7 ($\alpha = 0.7$), the execution success ratio of *CRAA* is 17% more than that using *MCMSA*. *MCMSA* jointly considers both mobile device users and mobile cloud resource providers; *CRAA* mainly optimizes the revenues of mobile cloud providers, it has better revenues than *OCRA* and *MCMSA*.

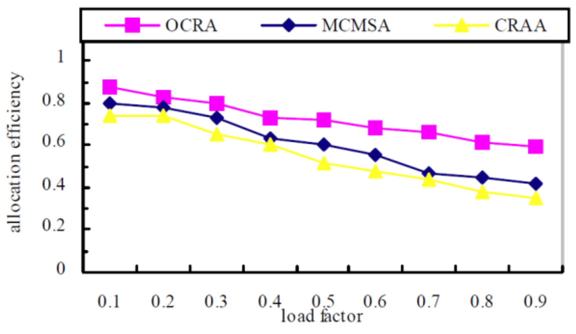

Figure 3. Allocation efficiency under different load factor

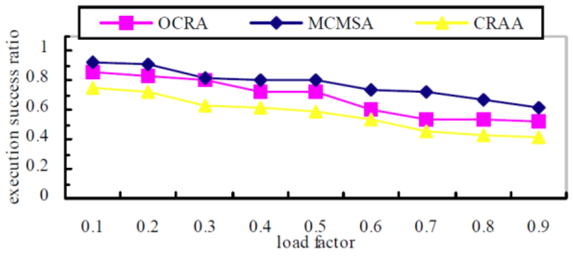

Figure 4. Execution success ratio under different load factor

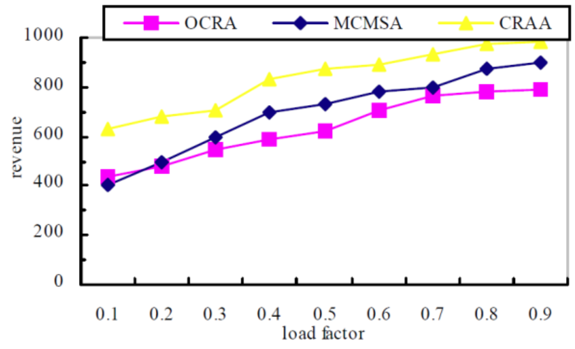

Figure 5. Revenue under different load factor

The deadline effects on allocation efficiency, execution success ratio and revenue are illustrated in Fig. 6-8. Fig. 6 shows the resource allocation efficiency with different deadlines. We can see that the resource allocation efficiency increases when the deadline increases. When the deadline is low, the job with low budget can not buy expensive cloud resource; this leads to low allocation efficiency. When the deadline is 350 (T=350), the resource allocation efficiency of *MCMSA* is 29% higher than T=100. Compared with *MCMSA*, the allocation efficiency of *OCRA* decreases more slowly than *MCMSA* when the deadline decreases. When deadline is 100 (T=100), allocation efficiency of *CRAA* decreases to 32%, allocation efficiency of *MCMSA* decreases to 61%. Fig. 7 is to show the effect of the deadline on execution success ratio. When the deadline is low, execution success ratios of *MCMSA*, *CRAA* and *OCRA* are low. When increasing deadline, execution success ratio of *MCMSA* outperforms *CRAA* and *OCRA*. Because under low deadline, more jobs can't be completed on time. When deadline is 100 (T=100), execution success ratio of *MCMSA* falls to 64% and execution success ratio of *OCRA* falls to 51%. From the results in Fig. 8, the revenue increases when the deadline decreases. When the deadline is low, the jobs need to be completed in short time, so mobile device user chooses more expensive cloud resources to process the jobs. However, when the deadline becomes higher, it is likely that the jobs can be completed before the deadline, so mobile device user considers using the cheaper cloud resources to complete jobs, the revenue of the mobile cloud provider becomes high. When the deadline is 300 (T=300), the revenue of *CRAA* is 17% higher than *OCRA* and 10% higher than *MCMSA*.

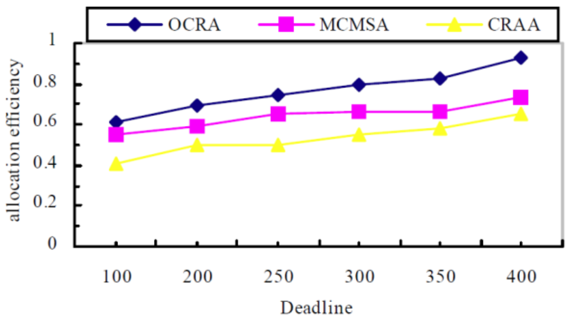

Figure 6. Allocation efficiency under different deadline

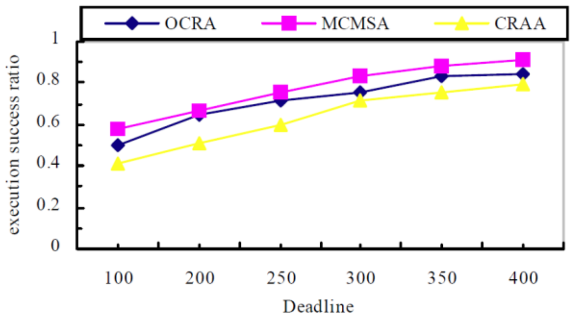

Figure 7. Execution success ratio under different deadline

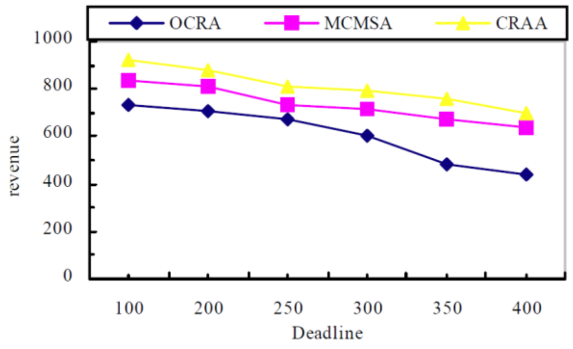

Figure 8. Revenue under different deadline

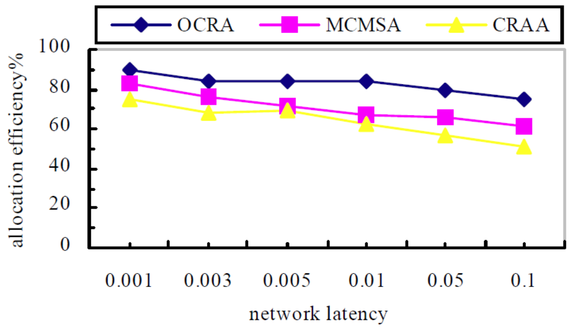

Figure 9. Effect of network latency on allocation efficiency

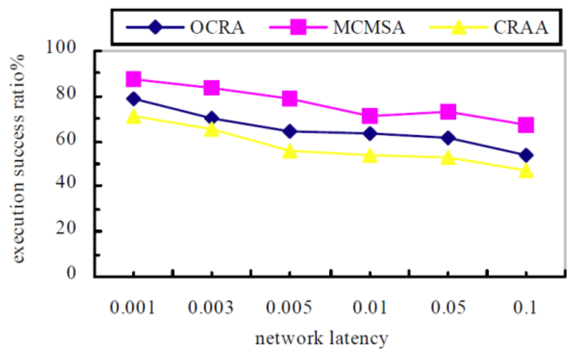

Figure 10. Effect of network latency on execution success ratio

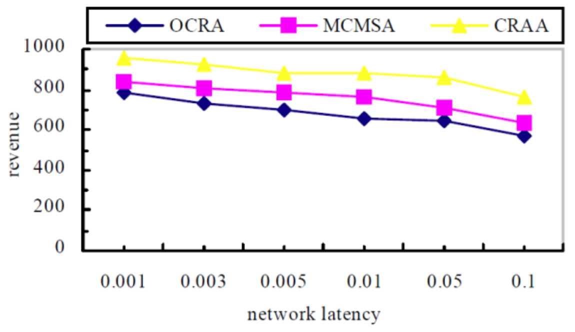

Figure 11. Effect of network latency on revenue

Figures 9 to 11 are to measure the effect of network latency on allocation efficiency, execution success ratio and revenue, respectively. Network latency refers to the time elapsed between the sending of a message to a router and the return of that message. Considering the resource allocation efficiency, from the results in Fig. 9, the X-axis shows a change in network, the resource allocation efficiency of *MCMSA* is as much as 18% less than *OCRA*. The resource allocation efficiency of *MCMSA* is as much as 17% higher than *CRAA*. When network latency reaches 0.05, the resource allocation efficiency of *MCMSA* is 19% more than *OCRA*. From the results in Fig. 10, lower network latency leads to higher execution success ratio. After network latency reaches 0.005, the execution success ratio of *OCRA* can be as much as 8% more than *MCMSA*. The execution success ratio of *CRAA* is as much as 17% less than *MCMSA*. The reason is that increasing network latency leads to longer times to complete tasks; so the execution success ratio becomes less. Fig. 11 shows the effect of varying network latency on the revenue. The revenue of *MCMSA* decreases when the network latency increases. Larger network latency enables mobile cloud user to expense more time for transfer and computation. When the network latency is large, the revenue is low. When the network latency is 0.05, the revenue of *MCMSA* is 19% less than the revenue when the network latency is 0.003. With the same network latency, *CRAA* can get more revenues than both *MCMSA* and *OCRA*.

The following experiments are to measure effect of different mobile cloud node numbers on the execution success ratio and allocation efficiency, respectively. Firstly, considering the execution success ratio, Fig. 12 shows that when the number of mobile clouds nodes increases up to 80, execution success ratio of *MCMSA* is as much as 17% less than that with *N*=10. The execution success ratio is larger when the number of mobile cloud nodes is smaller. The execution success ratio of *MCMSA* is higher than *OCRA* and *CRAA* when mobile cloud node increases. When the number of mobile cloud users is 100, the execution success ratio of *CRAA* decreases to 49%, the execution success ratio of *MCMSA* decreases to 64%, the execution success ratio of *OCRA* decreases to 56%.Considering allocation efficiency, as shown in Fig. 13, when the number of mobile cloud nodes increases, allocation efficiency deteriorates. When the number of mobile cloud nodes is 80, the allocation efficiency of *MCMSA* is as 34% less than the number of mobile cloud nodes is 10. Compared with *CRAA* and *MCMSA*, the allocation efficiency of *OCRA* slowly decreases than *CRAA* and *MCMSA* when the number of mobile cloud nodes increases. When the number of mobile cloud nodes is 100, allocation efficiency of *OCRA* decreases to 73%, resource allocation efficiency of *CRAA* decreases to 51%, allocation efficiency of *MCMSA* decreases to 59%. When the numbers of mobile cloud nodes are same, *OCRA* can get better allocation efficiency than both *MCMSA* and *CRAA*.

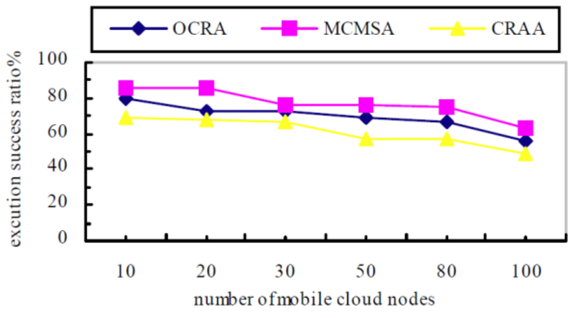

Figure 12. Execution success ratio versus the number of mobile cloud nodes

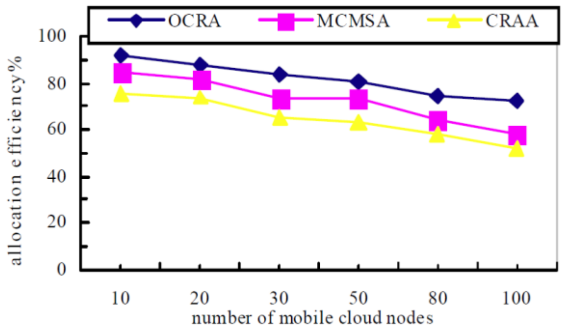

Figure 13. Allocation efficiency versus the number of mobile cloud nodes

From above simulation results, the objective of *MCMSA* is to satisfy mobile device users' needs, as well as optimize the profit of mobile cloud provider. *CRAA* aims to maximize the revenue of the mobile cloud service providers, *OCRA* aims to select the best resource allocation strategy in order to satisfy SLA in mobile cloud. The execution success ratio of *MCMSA* is better than *OCRA* and *CRAA*; *CRAA* behaves best in term of revenue; *OCRA* outperforms better than *MCMSA* and *CRAA*.

5. An Application Example

In this section, we take multimedia search as an example in mobile cloud environment, and apply the proposed multistage scheduling method to mobile cloud environment. Multimedia data consist of files recorded on mobile devices, including videos, photos, and sound clips. They also encompass files stored on mobile devices for entertainment, such as music and movies. Multimedia search application would find photos, videos, or music files whose contents are similar to that of an input sample. The multimedia search application allows the cloud users to browse through videos and images stored on cloud datacenter and search by time, location, and quality. The agent based multimedia search model in mobile cloud is shown in Fig. 14.

In our method, several agents are used, namely, mobile cloud provider agents, mobile device user agents, intelligent service agent and mobile cloud scheduler agent which implements scheduling of batch applications for mobile cloud computing environment. Intelligent service agent is to provide multimedia search support for mobile device user agents. Mobile cloud provider agents and mobile device user agents act on behalf of mobile cloud provider and mobile cloud users. Mobile device user agents send the requests to intelligent service agents to find the needed videos or other multimedia resource in the mobile cloud and then mobile device user agent can get the multimedia data for mobile cloud users. The mobile cloud scheduler agent receives the request from mobile device users and schedules the request to suitable cloud resource nodes. The mobile cloud scheduler agent monitors the task requests from mobile device user agent. It receives the task requirements and puts them into the task queue. While the task queue is not empty, the mobile cloud scheduler agent starts the multistage scheduling algorithm (*MCMSA*).

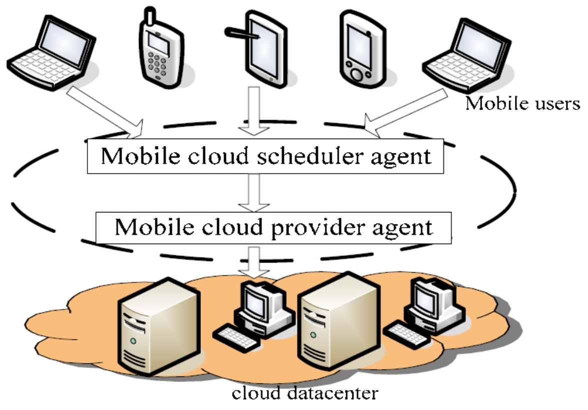

Figure 14. Agent based multimedia search model in mobile cloud

When the mobile cloud provider agent updates its price, it forwards the price to mobile device user agents; the mobile cloud resource price is put in a packet. Whenever the new price passes to mobile device user agent, it computes the benefit utility. According to the multistage scheduling algorithm, if the price is higher than the budget limit, mobile device user agent can't offer the payment for mobile cloud providers. The mobile device user agent can be informed the price for the next iteration. Searching some sort of multimedia resources in the mobile cloud requires cooperation between intelligent service agent and the mobile cloud provider agents. There are two kinds of cooperation strategies: agent based search strategy and message based search strategy. In agent based search strategy, intelligent service agent provides mobile cloud user agents with a list of choices of multimedia data located in different places, the mobile cloud user agent then checks these locations in turn (See Fig. 15).

In the message based search strategy (see Fig. 16), intelligent service agent conducts a search through the use of messages in order to select a suitable location to the mobile device user agent. Then the mobile device user agent can go to the destination location and avoid the high overhead of moving the mobile device user agent to all possible locations. The message based search strategy is essentially a search via direct communication between mobile device user agents and intelligent service agents.

6. Conclusions

Mobile cloud can facilitate the use of mobile devices to collect data, manipulate them and interact with scientific workflows running in the Cloud. The paper studies efficient multistage scheduling strategy for batch processing applications in mobile cloud. In

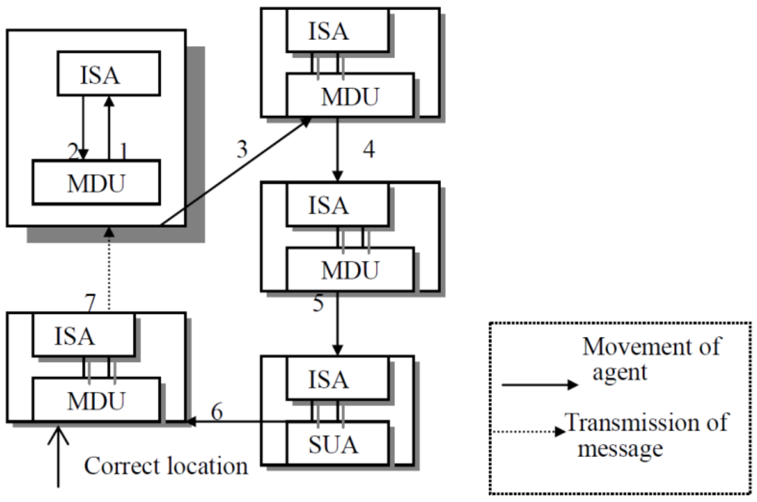

Figure 15. Agent based multimedia search strategy in mobile cloud (MDU=Mobile Device User Agent, ISA=Intelligent Service Agent)

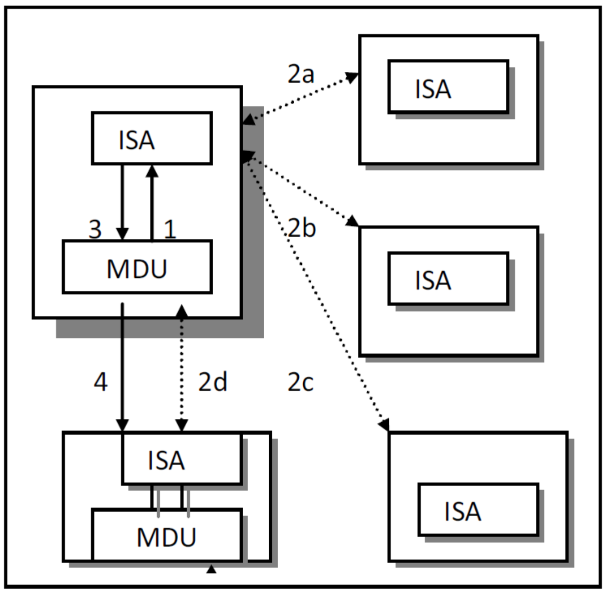

Figure 16. Message based multimedia search strategy in mobile cloud (MDU=Mobile Device User Agent, ISA=Intelligent Service Agent)

multistage scheduling model of mobile cloud, the objective of mobile device's batch applications optimization is to provision VMs for batch applications such that the mobile cloud utility is maximized subject to the resource constraints of mobile cloud datacenter and the requirements of mobile device's batch applications, respectively. In order to achieve a distributed solution, the mobile cloud multistage scheduling optimization is decomposed into divisible subproblems, which are processed in parallel by mobile device's batch applications and mobile cloud providers, respectively.

We take multimedia search as an example in mobile cloud environment, and apply the proposed multistage scheduling method to mobile cloud environment. The experiments aimed at comparing our mobile cloud multistage scheduling algorithm (*MCMSA*) with *CRAA* [2], which maximizes the revenue of the mobile cloud service providers and *OCRA* [13], which selects the best resource allocation strategy in order to satisfy SLA in mobile cloud. The objective of *MCMSA* is to satisfy mobile device users' needs, as well as optimize the profit of mobile cloud provider. So from the simulation results, execution success ratio of *MCMSA* is better than *OCRA* and *CRAA*; *CRAA* behaves best in term of revenue; allocation efficiency of *OCRA* is better than *MCMSA* and *CRAA*. In the future, we will move our mobile cloud multistage scheduling to real mobile cloud environment to test the feasibility and correctness. We also want to build mobile cloud platform for our campus.

Acknowledgements

The authors thank the editors and the anonymous reviewers for their helpful comments and suggestions. The work was supported by the National Natural Science Foundation (NSF) under grants (No.61472294, No.61171075), Key Natural Science Foundation of Hubei Province (No.2014CFA050), National Key Basic Research Program of China (973 Program) under Grant No.2011CB302601, Program for the High-end Talents of Hubei Province, and Open Fund of the State Key Laboratory of Software Development Environment (SKLSDE). Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

References

[1] **P. Bahl, Y. Han R, E. Li L, S. Mahadev.** Advancing the state of mobile cloud computing. In: *Proceedings of the third ACM workshop on Mobile cloud computing and services*. ACM, 2012, pp. 21-28.

[2] **R. Kaewpuang, D. Niyato, P. Wang, E. Hossain.** A Framework for Cooperative Resource Management in Mobile Cloud Computing. *IEEE Journal on Selected Areas in Communications*, 2013, Vol. 31, No. 12, 2685-2700.

[3] **H. Wu, Q. Wang, K. Wolter.** Tradeoff between performance improvement and energy saving in mobile cloud offloading systems. In: *IEEE International Conference on Communications Workshops (ICC)*, IEEE, 2013, pp. 728-732.

[4] **H. Yamauchi, K. Kurihara, T. Otomo, Y. Teranishi, T. Suzuki, K. Yamashita.** Effective distributed parallel scheduling methodology for mobile cloud computing. In: *Proceedings of the 17th Workshop on Synthesis and System Integration of Mixed Information Technologies (SASIMI'12)*, 2012, pp. 516-521.

[5] **X. Lin, Y. Wang, M. Pedram.** An optimal control policy in a mobile cloud computing system based on stochastic data. In: *2nd International Conference on Cloud Networking (CloudNet)*, IEEE, 2013, pp. 117-122.

[6] **S. Abolfazli, Z. Sanaei, A. Gani, M. Shiraz.** MOMCC: Market-Oriented Architecture for Mobile Cloud Computing Based on Service Oriented Architecture. In: *IEEE International Conference on.Communications, China Workshops (ICCC)*, 2012, pp. 8-13.

[7] **Z. Shi, R. Gu.** A framework for mobile cloud computing selective service system. *Wireless Telecommunications Symposium (WTS)*, 2013, pp. 1-5.

[8] **J. S. Park, E. Y. Lee.** Entropy-based grouping techniques for resource management in mobile cloud computing. In: *Ubiquitous Information Technologies and Applications*, Springer Netherlands, 2013, 773-780.

[9] **T. Nishio, R. Shinkuma, T. Takahashi, B. N. Mandayam.** Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud. In: *Proceedings of the First International Workshop on Mobile Cloud Computing & Networking*, ACM, 2013, pp. 19-26.

[10] **M. Shiraz, S. Abolfazli, Z .Sanaei, A. Gani.** A study on virtual machine deployment for application outsourcing in mobile cloud computing. *The Journal of Supercomputing*, 2013, Vol. 63, No. 3, 946-964.

[11] **P. Balakrishnan, C. K. Tham.** Energy-Efficient Mapping and Scheduling of Task Interaction Graphs for Code Offloading in Mobile Cloud Computing. In: *Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing*, 2013, pp. 34-41.

[12] **M. H. Mohammad, H. M. Shamim, J. Sarkar.** Cooperative game-based distributed resource allocation in horizontal dynamic cloud federation platform. *Information Systems Frontiers*, published online.

[13] **P. P. Hung, T. Bui, M. A. G. Morales, M. V. Nguyen, E. Huh**. Optimal collaboration of thin–thick clients and resource allocation in cloud computing. *Personal and Ubiquitous Computing*, 2014, Vol. 18, No. 3, 563-572.

[14] **L. Yang, J. Cao, S. Tang, Y. Yuan.** A framework for partitioning and execution of data stream applications in mobile cloud computing. *ACM Sigmetrics Performance Evaluation Review*, 2013, Vol. 40, Nr. 4, 23-32.

[15] **D. Li, J. Cao, X. Lu.** Efficient Range Query Processing in Peer-to-Peer Systems. *IEEE Transactions on Knowledge and Data Engineering*. 2009, Vol. 21, No. 1, 78-91.

[16] **Z. Sanaei, S. Abolfazli, A. Gani, M. Shiraz.** SAMI: Service-based arbitrated multi-tier infrastructure for Mobile Cloud Computing. *1st IEEE International Conference on Communications in China Workshops (ICCC)*, 2012, pp. 14-19.

[17] **S. Sindia, A. S. Lim, S. Gao, V. Agrawal, B. Black, P. Agrawal.** MobSched: Customizable scheduler for mobile cloud computing. In: *IEEE 45th Southeastern Symposium on System Theory (SSST)*, 2013, pp. 129-134.

[18] **Y. Zhang, D. Niyato, P. Wang.** An auction mechanism for resource allocation in mobile cloud computing systems. In: *Wireless Algorithms, Systems, and Applications.* Springer, 2013, 76-87.

[19] **J. Park, H. Yu, E. Y. Lee.** Resource allocation techniques based on availability and movement reliability for mobile cloud computing. In: *Distributed Computing and Internet Technology*. Springer, 2012, 263-264.

[20] **C. L. Li, L. Y. Li**. Phased Scheduling for Resource-Constrained Mobile Devices in Mobile Cloud Computing, *Wireless Personal Communications*, 2014, Vol. 77, No. 4, 2817-2837.

[21] **C. L. Li, L. Y. Li**. Optimal Resource Provisioning For Cloud Computing Environment. *Journal of Supercomputing*, 2012, Vol. 62, No. 2, 989-1022.

[22] **A. Andziulis, D. Dzemydienė, R. Steponavičius, S. Jakovlev**. A Robust Intelligent Construction Procedure for Job-Shop Scheduling. *Information Technology and Control*, 2014, Vol. 43, No. 3, 217-229.

Received June 2013.