Improving map reduce task scheduling and micro-partitioning mechanism for mobile cloud multimedia services

S. Saravanan*

Department of CSE,
M. Kumarasamy College of Engineering,
Karur, Tamil Nadu, India
Email: jeyasaraa@gmail.com
*Corresponding author

V. Venkatachalam

The Kavery Engineering College,
Salem, Tamil Nadu, India
Email: vv01062007@hotmail.com

Abstract: Over the several decades, there is a massive improvement in the computer technology which leads to infinite number of resources in all over the world. Many computing devices have to generate data that comes from various domains. In order to reduce the time complexity and the storage space of data, a novel technique namely map reduce programming model has been proposed to divide the workload among computers in a network to enhance the performance. To rectify the challenging issue of uneven data distribution, and also to enhance the process of load balancing along with memory consumption of computer, data sampling is highly preferred. To enhance the accuracy in scheduling, an innovative method called map reduce task scheduling algorithm is proposed for job deadline constraints. This algorithm classifies the nodes into several levels in heterogeneous clusters. Under this algorithm, a novel data distribution model has been elucidated in which it distributes data according to the node's capacity level respectively.

Keywords: MTSD; map reduce; workload distribution.

Reference to this paper should be made as follows: Saravanan, S. and Venkatachalam, V. (2016) 'Improving map reduce task scheduling and micro-partitioning mechanism for mobile cloud multimedia services', *Int. J. Advanced Intelligence Paradigms*, Vol. 8, No. 2, pp.156–167.

Biographical notes: S. Saravanan is working as an Assistant Professor of Computer Science and Engineering Department at M. Kumarasamy College of Engineering (Autonomous), Karur, Tamil Nadu. He received his Master degree in Computer Science and Engineering in 2010 and BE in Computer Science and Engineering in 2006 at Anna University, Chennai, Tamil Nadu. He is interested in cloud computing and web services.

V. Venkatachalam is working as the Principal in The Kavery Engineering College, Salem, Tamil Nadu, India. He has 17 years of experience. He is interested in cloud computing and web services.

This paper is a revised and expanded version of a paper entitled 'Improving map reduce task scheduling and micro-partioning mechanism for mobile cloud multimedia services' presented at the ICICDS2014, School of Computing, Sastra University, 12–13 September 2014.

1 Introduction

Currently, a large number of mobile cloud platforms have been proposed to provide a pervasive computing environment based on cloud recommendation system (Hu et al., 2012; Rodrigues et al., 2012; Zhu et al., 2011; Liu et al., 2011). In most of them, each mobile device is linked with a system level service in the cloud infrastructure. Moreover, with the rapid development of wireless communication technologies, users are expected to use more multimedia services in the mobile cloud to avoid the installation of the software in mobile devices. Informally, it is called mobile cloud multimedia services (MCMSs). Clearly, when all the multimedia services move to the cloud, MCMS becomes larger and more complicated, thus it is necessary to design an efficient scheduling scheme that dynamically allocates appropriate user service requests to available multimedia servers without the help of a centralised controller (Hu et al., 2012; Rodrigues et al., 2012; Wen et al., 2012; Meng et al., 2011; Zhu et al., 2011).

Basically, since various service requests usually come from different users and the scheduling policies for MCMS are typically delay sensitive, intuitively, the users are allocated to the servers with less service time to reduce transmission and waiting delays. However, in this case, the faster servers naturally become busy since they possess less free time than the slower ones, and hence, this leads to significantly raise the energy consumption (Teng et al., 2011). More recently, green multimedia services attracted so much attention and became an irreversible trend. As a result, it is necessary to enable all the servers to have the same (or similar) free time when assigning heterogeneous services. This encourages us to design an exquisite scheduling policy for MCMS with the following goals: minimising the delay of the service, and achieving impartial free time among the servers regardless of their service time.

Although substantial scheduling schemes have jointly taken into account the delay and energy in various cloud environments, a key point assumed in most existing works is that average service time and request rate, are known for the system operator. Obviously, this is helpful for simplifying the underlying scheduling problem and constructing easy service models (Teng et al., 2011). Nevertheless, this assumption cannot be always satisfied in practical MCMS. It should be noted that even though the operator utilises a powerful prediction technology with a sufficient former data, the above two parameters, in particular for the mobile cloud, are unknown as well. As a result, it is meaningful to study the scheduling problem when the important parameters are unavailable. Specifically, we informally define this case as blind scheduling. Furthermore, since the multimedia services sometimes are real-time, the scheduling policies should be implemented online for practical uses.

Essentially, blind online scheduling results in some critical difficulties:

- designing the user routing according to the availability and ability of the servers
- determining the server assignment without knowing the service request and time
- implementing the scheduling in a distributed way for online operation.

These three problems are interacted with each other, and hence they should be resolved jointly rather than separately. Our objective in this work is to jointly consider the above three problems by designing a total blind online scheduling algorithm (BOSA) for the general MCMS. Specifically, we consider that new users are routed to the server whose free time is the largest for achieving the energy efficiency. Then, the available servers are assigned based on a separated method to get the delay as small as possible. Moreover, the allocation can be set easily for the online implementation.

The rest of this article is organised as follows. First, overview the related works on scheduling in cloud environments. Then provide the system model and formulate the problem. I design afterwards a blind online scheduling scheme and show its main characteristics, and apply the proposed scheduling scheme; subsequently, we conduct numerical simulations to demonstrate the efficiency of the proposed scheme. Finally, conclude the article with a summary.

2 Related work

To improve the resource utilisation of the mobile cloud, various scheduling mechanisms have been developed in recent years. In general, they can be broadly divided into two categories:

- implementing the scheduling with predictable cloud parameters
- considering scheduling with unknown user request rate and server service rate.

Specifically, for the first one, the majority of the current literature is based on approximations of user request by complete trial and error or by manual settings. In particular, some studies use the servers approximation technique to achieve the near optimal solution (Hu et al., 2012; Rodrigues et al., 2012; Wen et al., 2012); Meng et al. (2011) and Zhu et al. (2011) focus on heterogeneous multimedia services with different QoS requests by separating the user request and service ability. In addition, variable approximate techniques are applied to the scheduling scheme with multiple goal functions to achieve the trade-off between various performance metrics, e.g., transmission delay, power consumption, etc. Recently, optimal scheduling in heavy traffic regimes has received considerable attention. Interestingly, the correlation among the request rate, the service time, and the energy consumption is so small and even ignorable in this case. As a result, these metrics can be treated independently and the joint algorithm can be designed individually (Teng et al., 2011).

In fact, the uncertainties of the request rate and service rate further complicate the scheduling problem. Specifically, Verma et al. (2011) first derives the relationship between the service request rate and the user waiting time. Subsequently, a vast volume of papers concentrated on the performance analysis with suboptimal performance on the

service time or the energy consumption. Note that the majority of existing blind scheduling schemes are based on the stochastic optimisation, which typically undergoes tremendous computation complexity. Therefore, these scheduling schemes can hardly be implemented online (Teng et al., 2011). Most recently, Liu et al. (2011) utilise a stochastic model to predict the service request rate, but its complexity is high for the online operation. The study of heterogeneous users with different QoS requirements and unknown request rates is conducted in a sequence of papers (Chen et al., 2010; Zhou and Wang, 2010; Hammoud, 2010), in which the underlying relationship between the above two parameters are clearly formulated and how the parameters impact the performance is well investigated. In this work, we will further extend these works by proposing a simple blind scheduling scheme for online operation, and at the same time, we inherit the properties of the previous work (Teng et al., 2011).

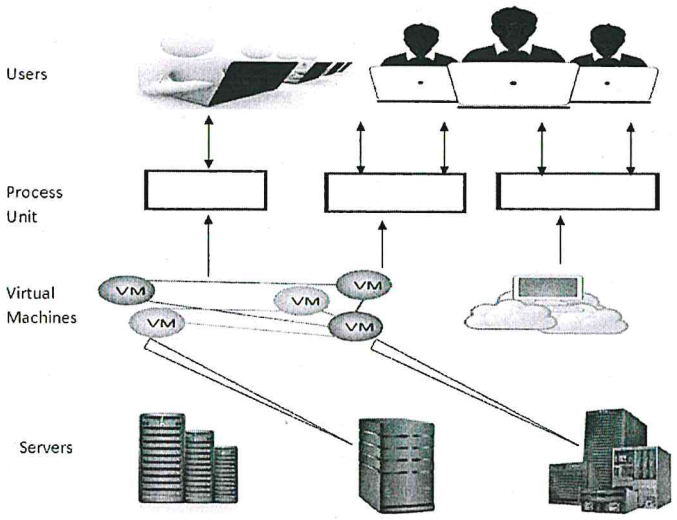
It is important to note that although the current work is derived from Teng et al. (2011), there are two distinguished differences:

- The scheduling algorithm in this work is much simpler, which facilitates the online operation in the mobile cloud.
- The condition to achieve the asymptotical optimality is more critical in this work, but it can also be held in the heavy traffic regime. Details of this will be discussed later.

3 MCMS architecture

In this section, we describe a general MCMS architecture based on a pervasive multimedia service framework. Inspired by the concept of the separated function, MCMS is represented by four parts which are shown in Figure 1. Specifically, mobile users request heterogeneous multimedia services to the mobile clouds via base stations, e.g., access point, transceiver station, etc., which construct the user interfaces between the mobile cloud and the users. Mobile users' state information, e.g., ID and location are also sent to the process unit to decide whether to agree or decline the request (Verma et al., 2011). Note that in the architecture of MCMS, each multimedia service is encoded independently with the utility computing, visualisation, and service-oriented link (e.g., heterogeneous multi-media applications, database servers, etc.). Basically, the process unit plays the role of accessing control, and it usually consists of eight elements: access mode, service provider information, user request information, unified interface, energy consumption, flow scheduler, resource allocation, and user state information. In particular, the energy consumption, flow scheduler, and resource allocation strategy can be integrated into a general scheduling middle-ware which can be embedded into the process unit. In this work, we will concentrate on this part by designing a simple scheduling scheme for online operation. After that, the service requests are delivered to virtual machines (VMs) which act as the bridge between the service requests and available resource. In particular, the VMs first find appropriate servers which can provide the corresponding multimedia services, then assign the available server with a specific goal function (e.g., the minimum energy cost, the minimum waiting time, etc.).

Figure 1 Mobile cloud multimedia architecture (see online version for colours)



Based on the above analysis, the architecture of MCMS is a classical large-scale distributed system. From the function perspective, the architecture of MCMS can also be divided into four layers:

- *Server maintain (SM) layer*: This layer provides the hardware facility and infrastructure, in which multiple servers are connected with hybrid networks to provide multimedia services. Typically, these servers are placed in ideal locations with plentiful of power supply and low venture of breakdown.
- *Data mapping (DM) layer*: The mechanism of DM is built on top of the SM layer. DM acts as the connection between the physical server and QoS requests, and it involves the provision of servers, content update, and networking components. The users should pay for the service time and service quality, and the DM layer can considerably reduce the waiting time among the huge number of servers. Thus, users can save cost as the payment strongly depends on how much the server has been utilised. In particular, in the DM layer, the mapping fashion can be expanded or reduced dynamically according to the practical need (Hu et al., 2012; Zhu et al., 2011).
- *VM layer*: VM performs as a service platform which offers an advanced integrated environment for building, testing, and deploying user services. Currently, the VM techniques are quite mature, and almost all the clouds employ it to improve the QoS (Rodrigues et al., 2012).
- *Access control (AC) layer*: In this layer, users can request multimedia services and access the mobile cloud via the different kinds of networks, and they should pay for this access. For example, Microsoft's Live Mesh also allows sharing multimedia services across numerous mobile devices simultaneously.

Furthermore, combing the architecture of MCMS with the above four function layers, it is obvious that the MCMS is an open and integrated service platform, which provides different interfaces for different multimedia applications. In particular, MCMS has the following additional advantages compared to traditional mobile clouds:

- *Dynamic maintain*: Dynamic on-demand multimedia service maintain is an efficient and flexible way for multimedia service providers and mobile users to run their applications in a dynamic environment without additional resource requests.
- *Scalability service*: The deployment of mobile multimedia applications can be adapted and scaled to meet the unpredictable user QoS requests due to variable available resources. Multimedia application providers can easily involve an unexpected service without or with little extra resource usage.
- *Information sharing*: Multimedia application providers can share the resources and costs, in the open architecture of MCMS, to support numerous services and large number of users.
- *Information integration*: Service requests from a variety of users and available services from different multimedia application providers can be integrated easily in the framework of MCMS by employing dynamic maintenance, scalability service, and information integration (Chen et al., 2010).

4 Blind online multimedia scheduling

In this section, we first briefly describe the system model. Next, we propose a blind scheduling scheme by fully making use of the blind system information and large-scale cloud in heavy traffic. Additionally, we apply our blind scheduling scheme to a cloud-based program recommendation system.

4.1 System model

We consider a classical mobile cloud environment. Specifically, $U = \{1, \ldots, U\}$ represents user classes and $S = \{1, \ldots, S\}$ denotes server classes. Users request multimedia services according to an independent Poisson processes P_u, $u \in U$. Assume that service time of each server satisfies an exponential function, and the server from the same class $s \in S$ has the same average service time. Moreover, let γ be a specific scheduling scheme and $t \geq 0$ be a time slot, and $W_u(t; \gamma)$ denotes the waiting time of class $u \in U$ user that has the longest waiting time (Teng et al., 2011). By jointly considering waiting-time and energy consumption, an efficient scheduling should have both low delay for users and an impartial task division among the servers. To that end, we introduce $Y_s(t; \gamma)$ to show the minimum free time of server class s, and define an average factor

$$f_s = \frac{Y_s(t;\gamma)}{\Sigma_s Y_s(t;\gamma)} \tag{1}$$

Note that the average factor defined in this work is different from that of Teng et al. (2011) in which an average free time of server is used. There are two main reasons that we replace the previous one:

- the average value is difficult to obtain in a dynamic environment
- the average value sometimes cannot guarantee the server's fairness.

Let $T > 0$ be a given time period, which can be viewed as the service time of a specific multimedia application, e.g., $T = 90$ min for a football match. As we know, the average factor can be set directly with perfectly predictable system parameters, however, for blind scheduling; the average factor itself is an unknown variable depending on the realisation of the request rates. To circumvent this difficulty, we employ a probability-constrained formulation in which the specific constraint is set by some prespecified fraction of the parameters. Usually, the constrained probability is artificially set by the system operator, who determines the system performance on the trade-off between the system performance and complexity.

Essentially, the outline of the probability-constrained formulation is as follows: the system operator first subjectively chooses a venture level η, which represents the maximum violated value of the average factor. As we know, the advantage of the probability constrained formulation lies in elastically adapting to the venture level that the scheduler can tolerate. In general, blind scheduling aims to solve the following stochastic probability optimisation problem, where $\eta \in (0, 1)$ is the venture level, $P_u(T)$ shows the recorded user request over time period T, and $W_{u,k}$ represents the waiting time up to T by the k^{th} class u that has requested the service at the beginning of time.

In fact, the core problem of resolving the goal function is how to control the match error given a tolerable limit from a long-time scale, which can yield the desired asymptotical optimality.

$$\begin{aligned} &\min_{\gamma} \sum_{u=1}^{U} \sum_{k=1}^{P_u(T)} \frac{W_{u,k}(\gamma)}{P_u(\gamma)} \\ &\text{s.t.} \quad P\left(\frac{Y_s(t,\gamma)}{\Sigma_{\gamma^{s_\gamma}}(t;\gamma)}\right) \geq 1-\eta, \end{aligned} \tag{2}$$

Indeed, this mission is possible in the heavy traffic regime in which the routing and assignment are independent of each other (Zhou and Wang, 2010). To be more specific, the routing algorithm first takes charge of the service time, then controls the practical waiting time for a user requesting a specific multimedia service, while the assignment algorithm is responsible for distributing the servers into different classes according to different multimedia service requests.

Moreover, we are interested in asymptotical optimality criteria in which determines the queue length and waiting time, rather than the system service state (e.g., service time). If the system service time is set as the asymptotical optimality criteria, then, from the mathematical perspective, any blind policy, e.g., first-come-first-served, etc., cannot resolve the goal function due to the definitions of P_u and $W_{u,k}$. Moreover, the formulation

of the goal function concentrates on an important relationship between the service time and average factor. In particular, each server has its own queue, and the decision on which user should be allocated to a queue is made as soon as the user arrival. In this regard, this strategy implies that a well-defined separated policy is a possible solution because in this case the system operation area belongs to the heavy traffic regime. It is important to note that the asymptotically optimal allocation should grow linearly as the increase of the user request rate. This based on the following two considerations:

- linear function is easy to calculate in an online fashion
- linear function can well fit the user request rate in a heavy traffic (Meng et al., 2011).

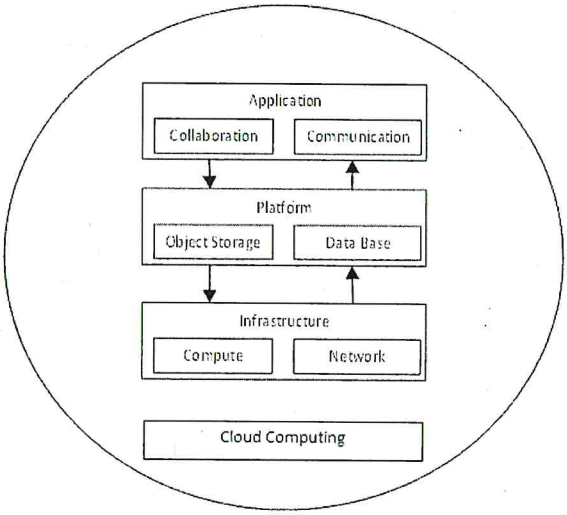
Our previous work, Teng et al. (2011) have demonstrated that the upper and lower allocation bounds can be used to achieve an asymptotic optimality. In this work, we extend Teng et al. (2011) to an online implementation. To this end, one should calculate the allocation values in a very simple way.

4.2 Blind scheduling strategy

Based on the above discussion, we can design a simple blind online scheduling scheme by jointly considering routing and assignment but separately developing them. Specifically, we assign available multimedia servers based on the last time-slot information of the users' requests, and route the heterogeneous multi-media flows according to the first-come-first-served rule. We propose a BOSA, it is obvious that the proposed blind scheduling algorithm (BSA) is suitable for practical use in the sense that it does not require the knowledge of the system parameters such as the request rates, service rates, etc. Instead, at each time-slot, one only needs to calculate s^* and u^* with computational complexity $O(1)$ which can be implemented online in a distributed way. The separated approach is attractive because the computation of s^* and u^* that are required for a reasonable approximation grows only linearly with the dimension of users. Moreover, the results can be used to derive estimates on the system parameters with a desired level of accuracy, and this point is consistent with our probability-constrained formulation.

Next, we employ the above scheduling scheme in a practical mobile cloud environment, which is shown in Figure 2. In general, there are two distinct types: one is to add the algorithm in the virtualisation part which attains an approximate solution of the goal function within the cloud, and the other one is to implement it in the task division and mapping part which act as a middle-ware between the cloud virtualisation and multi-media integration. For the former, it focuses on the cloud resource, providing QoS in the cloud infrastructure to support heterogeneous multi-media services with unknown user demand and available server. For the latter, it aims at improving cloud QoS in the upper layers, such as QoS in the application layer and QoS mapping between the cloud infrastructure and the QoS request. For ease of implementation, we employ the first one in our simulation. Note that, in fact, the results of the second one are able to achieve the same performance as that of the first type.

Figure 2 Mobile cloud environment



5 Numerical results

In this section, we evaluate the performance of the BOSA by conducting simulation experiments in a practical environment, which consist of three kinds of users and servers: audio, file, and video ($U = S = 3$). We compare our blind online scheme with a full-information case and the BSA in Teng et al. (2011). For the full-information case, the scheduler knows the user request rate and server service time. For comparison purposes, we set $N = 90$, $N_s = 30$, $f_s = 1/S$ for $s \in [1, S]$, and $\eta = 90\%$. In order to capture the heavy traffic regime, we set the average request rate for each user to 9,000/second.

We first test the performance of the BOSA. Figure 4 shows the performance of our proposed BOSA versus those of full-information case and BSA. From the given results, we can observe that when the waiting time goes large, the performance gap between BOSA and full information is negligible. That is to say, BOSA achieves asymptotically optimal in the heavy traffic regime, which is consistent with our theoretical analysis. Moreover, Figure 4 also verifies that when the system traffic belongs to a heavy traffic regime, the processes of routing and assignment can be separately designed. In addition, from the given results, it is clear that the performance of BOSA is better than that of BSA. That is because the computational complexity of BOSA is much lower than that of BSA, and the scheduling scheme costs fewer handling time. As a result, the average waiting time of the user is lower.

Figure 3 The mechanism of including the map-reduce algorithm into BOSA (see online version for colours)

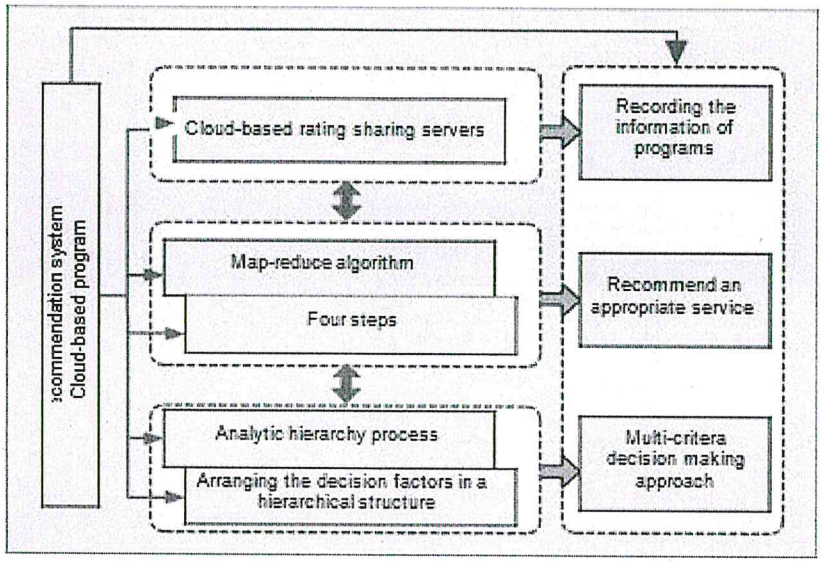
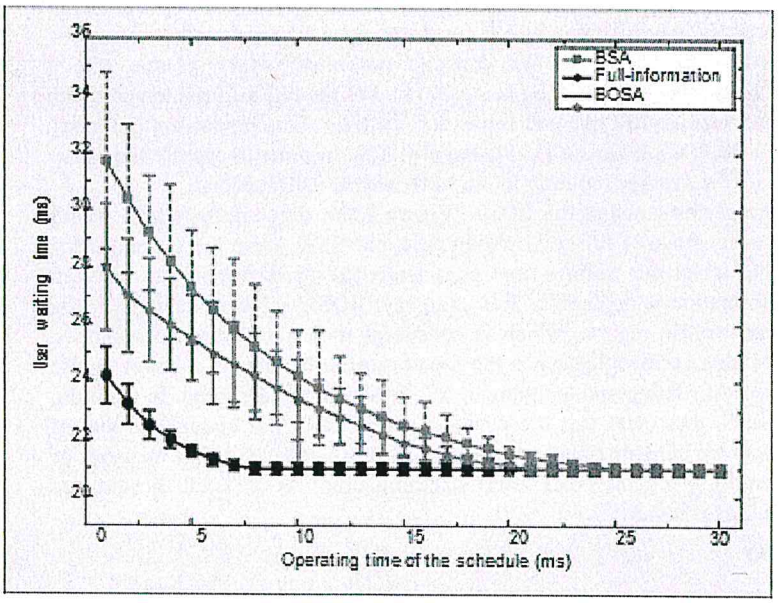
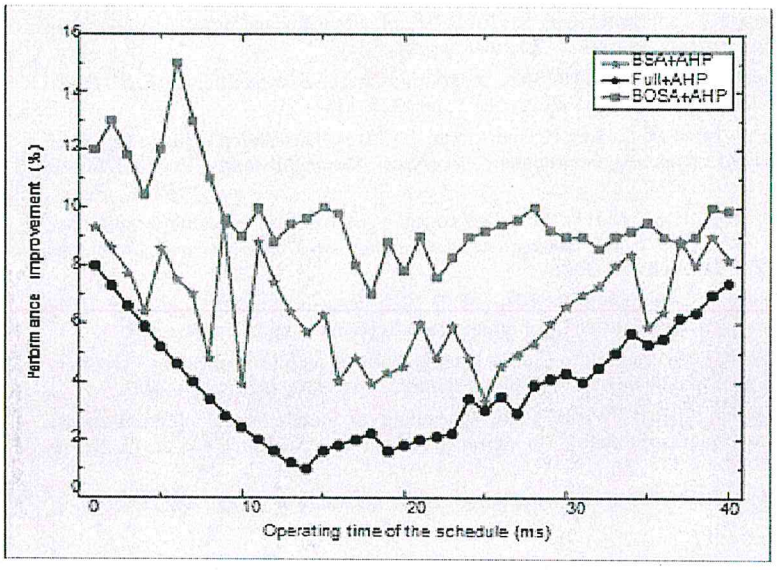


Figure 4 The performance comparison in terms of user waiting time (see online version for colours)



Next, we examine the performance improvement when employing the content-based program recommendation technique. We denote the three scheduling schemes by 'Full+AHP', 'BSA+AHP', and 'BOSA+AHP', respectively. Figure 5 exhibits the performance of each scheme when the operating time of the scheduling varies from 1 to 40 ms. Obviously, 'BOSA+AHP' achieves the best performance. That is because AHP utilises the complete comparison enabling the server to determine the trade-offs among criteria, and this mechanism is in accordance with the structure of the BSA and BOSA. That is, AHP can be embedded naturally and seamlessly in BSA and BOSA. Moreover, since BOSA just needs the information of $Y_s(t)$ and $W_u(t)$ at each time slot, AHP can be implemented at each server as well. That is the reason why BOSA also outperforms BSA in terms of the user waiting time.

Figure 5 Performance improvement when employing the content-based program recommendation technique (see online version for colours)



6 Conclusions

In this article, we consider a practical mobile cloud environment where the user waiting time and server service time are unknown. Our main contribution is to design a blind online scheduling scheme by jointly considering delay and energy among the servers. Specifically, we assign available multimedia servers based on the last time-slot information of the users' requests, and route the heterogeneous multimedia flows according to the first-come-first-served rule. Furthermore, we apply the blind scheduling scheme to a content recommendation system, and provide the detailed implementation steps. Extensive simulation results indicate that the proposed scheme can efficiently schedule heterogeneous multimedia flows to satisfy dynamic QoS requirements in a practical mobile cloud.

Acknowledgements

I would like to thank my colleagues and my guide Dr. V. Venkatachalam who have supported me towards to complete this paper successfully.

References

Chen, M. et al. (2010) 'Software agent-based intelligence for code-centric RFID systems', *IEEE Intelligent Systems*, Vol. 25, No. 2, pp.12–19.

Hammoud, S. (2010) 'MRSim: a discrete event based MapReduce simulator', in *2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery*, IEEE, August, pp.2993–2997.

Hu, G.Q., Tay, W.P. and Wen, Y.G. (2012) 'Cloud robotics: architecture, challenges and applications', *IEEE Network*, Vol. 26, No. 3, pp.21–28.

Liu, Y., Li, M., Alham, N.K. and Hammoud, S. (2011) 'HSim: a MapReduce simulator in enabling cloud computing', *Future Generation Computer Systems*, May.

Meng, S., Liu, L. and Wang, T. (2011) 'State monitoring in cloud datacenters', *IEEE Trans. Knowledge and Data Engineering*, Vol. 23, No. 9, pp.1328–1344.

Rodrigues, J., Zhou, L., Mendes, L., Lin, K. and Lloret, J. (2012) 'Distributed media-aware flow scheduling in cloud computing environment', *Computer Communications*, Vol. 35, No. 1, pp.1819–1827.

Teng, F., Yu, L. and Magoulès, F. (2011) 'SimMapReduce: a simulator for modeling MapReduce framework', in *2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering*, IEEE, June, pp.277–282.

Verma, A., Cherkasova, L. and Campbell, R.H. (2011) 'Play ItAgain, SimMR!', in *2011 IEEE International Conference on Cluster Computing*, IEEE, September, pp.253–261.

Wen, Y.G., Zhang, W.W. and Luo, H.Y. (2012) 'Energy-optimal mobile application execution: taming resource-poor mobile devices with cloud clones', *Proc. IEEE INFOCOM 2012*.

Zhou, L. and Wang, H. (2010) 'Toward blind scheduling in mobile media cloud: fairness, simplicity, and asymptotic optimality', to appear in *IEEE Trans. Multimedia*, Vol. 15, No. 4, pp.735–746.

Zhu, W. et al. (2011) 'Multimedia cloud computing', *IEEE Signal Proc. Mag.*, Vol. 28, No. 3, pp.59–69.