A QoS-aware Task Allocation Model for Mobile Cloud Computing

Mohammad Hossein Zarei
Router Lab., School of Electrical and Computer Engineering,
University of Tehran, Tehran, Iran.
mhzarei@ut.ac.ir

Milad Azizpour Shirsavar
Router Lab., School of Electrical and Computer Engineering,
University of Tehran, Tehran, Iran.
shirsavar@alum.sharif.edu

Prof. Nasser Yazdani
Router Lab., School of Electrical and Computer Engineering,
University of Tehran, Tehran, Iran.
yazdani@ut.ac.ir

***Abstract*— Mobile Cloud Computing (MCC) tries to offload computation from mobile devices, including smartphones and tablets, to cloud providers to solve the limitations of mobile devices with the power of cloud computing. Providing cloud computing quality of service (QoS) in mobile devices is harder to achieve compared to fixed devices due to the limited computing resources, storage and energy in mobile devices. On the other hand, node mobility causes rapid changes in network topology. Tasks in cloud computing infrastructure are performed with different quality of service requirements. Using appropriate allocation of tasks to cloud computing resources, can improve QoS. In this paper, we formulate the task allocation problem as a minimum nonlinear optimization problem. A randomized algorithm is proposed in which the result is close to the optimal solution. It improves the QoS in mobile devices by choosing suitable cloud providers for a given user application. The results show that the proposed algorithm is faster and more scalable compared to optimal solution.**

***Index Terms*— Mobile Cloud Computing, Quality of Service, Optimization Problem, Randomized Algorithm.**

I. INTRODUCTION

The advances in mobile devices have provided myriad of applications that enables users to do things that previously were done only with their personal computers.

Users are interested in mobile devices to be able to carry out heavy processing operations as well as personal computers. Nevertheless, these devices have major challenges such as limited processing resources, low storage and high energy consumption which refrains users from doing certain things[1].

In recent years, the technique of offloading heavy computation to run on supercomputers has been proposed to meet the limits on personal computers. Cloud computing scheme, could obviate some of the mentioned limitations [2]. In addition to having limited access to the Internet in terms of cost and speed, mobile devices also operate on limited battery power. They consume energy sometimes even more than the need to run the intended application, since they need more energy in order to transfer the load to a cloud provider [3]. The QoS provided by a single cloud is different according to their location and their free resources. So we need to make good decisions about what task and when the task should move to the cloud to reduce costs and provide user satisfaction. So, the correct choice based on the type of the task would improve the quality of service [5].

Mobile device users have different needs and suit their needs with different applications running on their device. Mobile devices applications that take advantages of cloud computing services can include the following: business, learning, healthcare, military, and entertainment. In general, security, low bandwidth, storage and heavy computational requirements are challenges that mobile cloud computing is attempting to solve it [1].

QoS requirements may vary according to the task. In Table I [6], some of the tasks along with their QoS requirements are listed. For example, in video streaming applications, with a delay less than 150 millisecond, jitter less than 100 milliseconds, throughput between 0.5 – 8 Mbps and less than a thousandth of a percent packet loss, it can be concluded that

users watching video will get good service and if this value is exceeded, the QoS will degrade.

QoS guarantees are important especially in networks with limited resources like mobile networks[7]. Task allocation to cloud computing resources plays an important role in the performance of the system. We formulate the task allocation problem and propose a randomized algorithm that will be discussed in detail in section III and IV.

The paper is organized in the following orders: In Section II, related works will be reviewed. Section III and IV explains the proposed algorithm and task allocation problem that is formulated as a minimum optimization problem. Experimental set ups and results are presented in Section V and conclusion of the study is given in Section VI.

II. RELATED WORK

QoS and task allocation in cloud and mobile cloud computing, since it involves user satisfaction and penalty issues for cloud providers, is very important and has been considered in recent research. Shakkeera *et al.* [8] propose an algorithm that runs on Cloudlet and tries to Consolidate applications that are idle to improve the QoS of new tasks and to reduce the energy consumption of the Cloudlet. The proposed algorithm also uses other mobile devices as a resource and runs applications on them. The proposed system minimizes the response latency, cost of application migration and it improves QoS like throughout and scalability among resources using load balancing techniques by MCC. However, the dynamic changes in the network and the possibility to run on other cloud providers are not considered.

A Cuckoo based allocation strategy [9] is proposed as an optimization problem with the aim of reducing the Makespan and the computational cost meeting the deadline constraints, with high resource utilization. The proposed approach is evaluated using CloudSim framework and the results indicate that the proposed model provides better QoS to the mobile cloud customers. The parameters of the communication network between the mobile device and the cloud, and also the dynamic network changes are not considered.

Saha *et al.* [10], introduce a task scheduling algorithm based on genetic algorithm using a queuing model to minimize the waiting time and queue length of the system. The proposed algorithm is compared with the first come first serve (FCFS) algorithm and simulation shows 20 percent improvement. The work gives less consideration to the network changes due to mobility of devices. Different QoS required by different tasks is not also considered in scheduling.

Zhao *et al.* [11] design a threshold-based policy to improve the QoS of MCC by cooperation of the local cloud and Internet cloud resources. Numerical results show that the QoS can be greatly enhanced with the assistance of Internet cloud when the local cloud is overloaded. Better QoS is achieved if the local cloud orders tasks according to their delay requirements, where delay-sensitive applications are executed ahead of delay tolerant applications. However, considering the deadline for requests, other QoS parameters and network topology changes are ignored.

ThinkAir [12] exploits the concept of smartphone virtualization in the cloud and provides method-level computation offloading. It focuses on the elasticity and scalability of the cloud and enhances the power of mobile cloud computing by parallelizing method execution using multiple virtual machine images. ThinkAir exploits mobile device virtualization to simultaneously execute multiple offloaded methods; thereby, reducing the application execution time. In ThinkAir, the decision-making process offloading is complex and the ability to run on different cloud providers with different characteristics is not considered.

III. TASK ALLOCATION PROBLEM FORMULATION

Since users runs various tasks on their mobile devices and cloud providers offer services with different quality, there needs to be a system that allocates tasks to the appropriate provider to improve the overall QoS of the system. Figure 1 shows the overall architecture of the system. The system checks the status of all service providers, then receives tasks, and assigns them to the appropriate provider.

A cloud provider can be hotspot Cloudlet and provides the service to mobile users with maximum bandwidth and minimum delay through WiFi network, but providers are faced with limited resources and cannot process all the requests. Also, not all service providers offer such services.

Mobile network operators also have cloud computing resources in their network infrastructure. Cellular network users can connect to these resources and run their applications on it. Based on the location of user, the quality of cellular network and congestion of operators' computing resources, a different QoS is provided to each mobile device. Operators' services have financial cost according to different policies. Public cloud providers like Google and Amazon are available via WiFi and cellular networks and different QoS and cost is provided according to the communication network and public cloud computing resources.

$$W_j^D + W_j^J + W_j^{PL} + W_j^{TR} + W_j^C = 1 \qquad \forall j \qquad (1)$$

In all equations, D, J, PL, TR and C represent the delay, jitter, packet loss, throughput and cost of the task, respectively. Given that the task *j* is transmitted through WiFi network or cellular network to the service provider *i*, different level of QoS is needed that is evaluated by function F. The function F in equations 2 and 3 are represented for WiFi and Cellular networks, respectively. The symbol *N* represents the normalized number of each QoS parameters. For example, ND_i^{wifi} shows the value of Normalized Delay for the provider *i* in WiFi network.

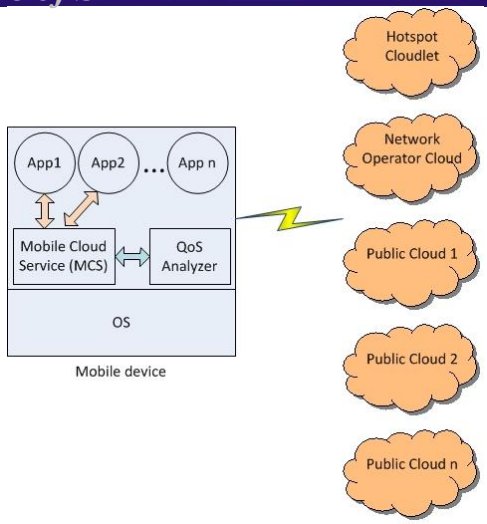

Fig. 1. The general architecture for quality of service evaluation in mobile cloud computing

$$F_{i,j}^{wifi} = W_j^{D} \times ND_i^{wifi} + W_j^{J} \times NJ_i^{wifi} + W_j^{PL} \times NPL_i^{wifi} + W_j^{C} \times NC_i^{wifi} - W_j^{TR} \times NTR_i^{wifi} \quad (2)$$

$$F_{i,j}^{cell} = W_j^{D} \times ND_i^{cell} + W_j^{J} \times NJ_i^{cell} + W_j^{PL} \times NPL_i^{cell} + W_j^{C} \times NC_i^{cell} - W_j^{TR} \times NTR_i^{cell} \quad (3)$$

Each task runs on only one cloud provider simultaneously. This issue is modeled with equation 4. $x_{j,i,w}$ and $x_{j,i,c}$ are binary variables which indicate that the task j will run on the provider i or not. w and c represent the WiFi and cellular networks, respectively. n represents the total number of tasks.

$$\begin{aligned} & x_{j,i,w} \in \{0,1\} \quad \forall j, \forall i \\ & x_{j,i,c} \in \{0,1\} \quad \forall j, \forall i \\ & \sum_{j=1}^{n}\sum_{i=1}^{p}\left(x_{j,i,w} + x_{j,i,w}\right) \le n \end{aligned} \quad (4)$$

Each task needs the minimum QoS requirement to run on cloud services that is represented by variable *Req*. The requirements for the task j to run on cloud provider i is modeled with equation 5 and 6 for the WiFi and cellular network, respectively.

$$\begin{aligned} & x_{j,i,w} \times D_i^{wifi} \le Req_j^{D} \\ & x_{j,i,w} \times J_i^{wifi} \le Req_j^{J} \\ & x_{j,i,w} \times PL_i^{wifi} \le Req_j^{PL} \\ & x_{j,i,w} \times C_i^{wifi} \le Req_j^{C} \\ & x_{j,i,w} \times Req_j^{TR} \le TR_i^{wifi} \times x_{j,i,w} \end{aligned} \quad (5)$$

$$\begin{aligned} & x_{j,i,c} \times D_i^{cell} \le Req_j^{D} \\ & x_{j,i,c} \times J_i^{cell} \le Req_j^{J} \\ & x_{j,i,c} \times PL_i^{cell} \le Req_j^{PL} \\ & x_{j,i,c} \times C_i^{cell} \le Req_j^{C} \\ & x_{j,i,c} \times Req_j^{TR} \le TR_i^{cell} \times x_{j,i,c} \end{aligned} \quad (6)$$

Considering the limitations of WiFi and cellular networks on bandwidth, task allocation problem is formulated with equation 7. This problem is more complicated example of the 0-1 knapsack problem and is NP-Hard, so cannot be solved in polynomial time, while mobile devices should allocate tasks with lower power consumption and in a short time. So, a randomized algorithm is presented in next section that the results are analogous to those for optimum solution.

$$\text{Minimize} \sum_{j=1}^{n}\sum_{i=1}^{p}\left(x_{j,i,w} \times F_{i,j}^{wifi} + x_{j,i,c} \times F_{i,j}^{cell}\right)$$

SubjectTo

$$\begin{aligned} & W_j^{D} + W_j^{J} + W_j^{PL} + W_j^{TR} + W_j^{C} = 1 \quad \forall j \\ & x_{j,i,w} \in \{0,1\} \quad \forall j, \forall i \\ & x_{j,i,c} \in \{0,1\} \quad \forall j, \forall i \\ & \sum_{j=1}^{n}\sum_{i=1}^{p} x_{j,i,w} Req_j^{TR} \le MaxTR_{Wifi} \\ & \sum_{j=1}^{n}\sum_{i=1}^{p} x_{j,i,c} Req_j^{TR} \le MaxTR_{Cell} \\ & \sum_{j=1}^{n}(x_{j,i,c} + x_{j,i,w}) Req_j^{TR} \le MaxTR_i \quad \forall i \\ & \text{Request constraints in equations 2 and 3} \end{aligned} \quad (7)$$

IV. PROPOSED ALGORITHM

Due to limitations of mobile devices and as the task allocation is an NP-Hard problem; we propose a randomized algorithm that allocates the task to cloud providers in

polynomial time with *O(pn)* time complexity where *p* and *n* indicates number of providers and number of tasks, respectively.

In proposed algorithm, first, all tasks are stored in a list named *UnallocatedTasks*. Then, the algorithm will enter a loop to allocate the tasks to providers. One task is randomly chosen from the *UnallocatedTasks* list. According to data from the QoS Analyzer, for the selected task, list of all cloud providers and the network that satisfy the requirements are extracted and evaluated with the function *PenaltyFunction* and are stored on the roulette wheel. Since the provider and the network that provides fewer penalties is a better choice, we reverse the roulette wheel to let the tasks with lower penalties, more likely to be chosen. Using the Roulette Wheel, one provider is randomly chosen. The while loop is eventually terminated when all tasks are allocated.

Algorithm 1 Random Algorithm for QoS-aware Task Allocation

```
function TASKALLOCATION(Tasks , Providers and
   Communication Network)
   UnallocatedTasks ← Tasks
   while UnallocatedTasks is not empty do
      T ← Randomly select a task
      UnallocatedTasks.Remove(T)
      SuitableProviders ← Find cloud providers and
         communication network that satisfy QoS
         constraint
      RouletteWheel ← {}
      for i ← 1 to Size(SuitableProviders) do
         RouletteWheel[i] ←
            PenaltyFunction( SuitableProviders[i] )
      end for
      RouletteWheel ← Reverse( RouletteWheel )
      P ← Select a cloud provider by RouletteWheel
      Allocate task T to cloud provider P
   end while
end function
```

In addition to having a polynomial running time and a near-optimal solution, the proposed algorithm prevents the best provider to be saturated by many tasks; so tasks are allocated in a distributed manner.

V. EXPERIMENTAL RESULTS AND ANALYSIS

In this section we will discuss the implementation and analysis of the proposed algorithm. Since that same works in the field of QoS in mobile cloud computing are different in terms of providers and provided services, so the proposed algorithm has been compared with the optimal solution. In Table I [6], QoS requirements for different tasks which can be performed in MCC services are listed. To evaluate the proposed algorithm, in the absence of a comprehensive experimental simulator for MCC, simulation is implemented using the Java programming language. We have developed cloud providers and tasks according to the parameters of Table I and used some network measurement tools such as Ping and Iperf. Simulation was done on an Intel Core i5 3.4 GHz machine with 8 GB of RAM. Then, with the simulation of Branch-and-Bound algorithm, which yields the optimal solution, and the proposed algorithm, the outputs of two algorithms are demonstrated and analyzed. In the simulations, we have taken 7 cloud providers, each with different features, based on their QoS requirements. The number of tasks is between 1 and 20 inclusive. Branch-and-Bound algorithm is used to obtain the optimal solution, and since the optimal method can allocate no more than nine tasks, considering more than nine tasks is ignored in the simulation. After 10, 100 and 1000 iterations of the proposed algorithm, results are compared with the optimal algorithm. First, the running time is compared in Table II. As the results show, with the increasing number of tasks, the optimal algorithm took a long time to find an answer, but the proposed algorithm took less time compared with the optimal solution.

TABLE I. QOS REQUIREMENTS OF MOBILE CLOUD COMPUTING TASKS [6]

Task	Delay (ms)	Jitter (ms)	Packet Loss (%)	Throughput (bps)
Upload file	Med	N/A	0	High
Download file	Med	N/A	0	High
Image processing	<150	<400	<0.1	5-19.6 K
Sound processing	<150	<400	<1	9.6-19.6 K
Query	<250	N/A	0	<1 K
Audio stream	<150	<100	<0.1	60-80 K
Video stream	<150	<100	<0.001	0.5-8 M
Game	<200	N/A	0	<1 K

TABLE II. RUNNING TIME (IN MILLISECONDS) OF TASK ALLOCATION ALGORITHMS, WITH 7 CLOUD PROVIDERS

Number of Tasks	Optimal solution	Proposed algorithm (with 10 iterations)	Proposed algorithm (with 100 iterations)	Proposed algorithm (with 1000 iterations)
1	1	0.038	0.177	0.765
2	1	0.039	0.191	0.781
3	13	0.042	0.176	0.78
4	65	0.046	0.181	0.783
5	269	0.051	0.184	0.79
6	793	0.055	0.187	0.79
7	4,068	0.059	0.191	0.794
8	46,795	0.063	0.192	0.792
9	635,406	0.063	0.193	0.793
10	-	0.065	0.194	0.792
11	-	0.07	0.195	0.794
12	-	0.077	0.196	0.798
13	-	0.094	0.196	0.806
14	-	0.108	0.197	0.808

15	-	0.113	0.2	0.814
16	-	0.115	0.199	0.813
17	-	0.116	0.202	0.82
18	-	0.117	0.202	0.825
19	-	0.119	0.203	0.821
20	-	0.121	0.204	0.826

On mobile devices with limited processing resources and energy, spending a long time to find the answer will result in high consumption of the battery and user dissatisfaction of the system. The running time of the proposed algorithm is dependent on the number of iterations of the algorithm; nevertheless it is fast and applicable on mobile devices.

To find how close is the answer to optimal solution, we use the sum of Valuation Function F on all tasks. Lower number indicates better answer. The results of this function are shown in Table III. It shows that the 100 repeats of the algorithm results in a close answer to the optimal solution. Due to the optimal allocation of resources to tasks, with the increase in tasks, other tasks use other resources which have lower quality. So, the total amount of evaluation function F increases.

TABLE III. COMPARISON OF VALUATION FUNCTION FOR TASK ALLOCATION ALGORITHMS, WITH 7 CLOUD PROVIDERS

Number of Tasks	Optimal solution	Proposed algorithm (with 10 iterations)	Proposed algorithm (with 100 iterations)	Proposed algorithm (with 1000 iterations)
1	0.01578	0.07695	0.01578	0.01578
2	0.01578	0.07296	0.01578	0.01578
3	0.01578	0.0643	0.0159	0.01579
4	0.01578	0.06582	0.01658	0.01609
5	0.01578	0.06396	0.01742	0.01705
6	0.01578	0.06155	0.0181	0.01813
7	0.01578	0.05915	0.01917	0.0192
8	0.01578	0.06463	0.02	0.02004
9	0.01578	0.06051	0.02077	0.02087
10	-	0.06597	0.02075	0.02151
11	-	0.06466	0.02201	0.02214
12	-	0.0575	0.02312	0.02303
13	-	0.0462	0.02385	0.02355
14	-	0.03052	0.02392	0.02385
15	-	0.02589	0.02467	0.0241
16	-	0.02544	0.02473	0.02464
17	-	0.02526	0.02598	0.02559
18	-	0.0269	0.0264	0.02666
19	-	0.0277	0.0277	0.02763
20	-	0.0284	0.02851	0.0285

VI. CONCLUSION

Different applications run on mobile devices that do different tasks. These applications can be performed on cloud computing as well. To perform tasks on a single cloud, there are WiFi and cellular networks which each provides different QoS. Different tasks also need different QoS requirements. The WiFi and cellular networks and also cloud providers have limited throughput, so choosing the right cloud provider, complicates the communication network.

In this paper, the task allocation problem is formulated as a minimum optimization problem and was shown that it is an NP-Hard problem, so cannot be solved in polynomial time to reach an optimal solution. Thus, a random algorithm with polynomial time was proposed that returns a near-optimal solution. Then to verify the validity and accuracy of the algorithm, the proposed algorithm with repetitions of 10, 100 and 1000, was compared with a branch-and-bound algorithm which yields the optimal solution. The results show that the proposed algorithm is faster and more scalable. The solution is also close to the optimal solution and is acceptable for mobile cloud computing systems.

REFERENCES

[1] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, “A survey of mobile cloud computing: architecture, applications, and approaches,” *Wirel. Commun. Mob. Comput.*, 2011.

[2] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, “Clonecloud: elastic execution between mobile device and cloud,” in *Proceedings of the sixth conference on Computer systems*, 2011, pp. 301–314.

[3] K. Kumar and Y.-H. Lu, “Cloud computing for mobile users: Can offloading computation save energy?,” *Computer (Long. Beach. Calif).*, vol. 43, no. 4, pp. 51–56, 2010.

[4] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, “Mobile Cloud Computing: A Survey, State of Art and Future Directions,” *Mob. Networks Appl.*, Nov. 2013.

[5] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, “Mobile Cloud Computing: A Survey, State of Art and Future Directions,” *Mob. Networks Appl.*, vol. 19, no. 2, pp. 133–143, 2013.

[6] Y. Chen, T. Farley, and N. Ye, “QoS Requirements of Network Applications on the Internet,” *Information-Knowledge-Systems Manag.*, vol. 4, no. 1, pp. 55–76, Jan. 2004.

[7] S. Abolfazli, Z. Sanaei, M. H. Sanaei, M. Shojafar, and A. Gani, “Mobile cloud computing: The-state-of-the-art, challenges, and future research.” Willeys & Sons, 01-Feb-2015.

[8] L. Shakkeera and L. Tamilselvan, “Energy-Aware Application Scheduling and Consolidation in Mobile Cloud Computing with Load Balancing,” in *Emerging Research in Computing, Information, Communication and Applications SE - 25*, N. R. Shetty, N. H. Prasad, and N. Nalini, Eds. Springer India, 2016, pp. 253–264.

[9] S. Durga, S. Mohan, J. Dinesh, and A. Aneena, "Cuckoo Based Resource Allocation for Mobile Cloud Environments," in *Computational Intelligence, Cyber Security and Computational Models SE - 50*, vol. 412, M. Senthilkumar, V. Ramasamy, S. Sheen, C. Veeramani, A. Bonato, and L. Batten, Eds. Springer Singapore, 2016, pp. 543–550.

[10] S. Saha, S. Pal, and P. Pattnaik, "A Novel Scheduling Algorithm for Cloud Computing Environment," in *Computational Intelligence in Data Mining—Volume 1 SE - 39*, vol. 410, H. S. Behera and D. P. Mohapatra, Eds. Springer India, 2016, pp. 387–398.

[11] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "A Cooperative Scheduling Scheme of Local Cloud and Internet Cloud for Delay-Aware Mobile Cloud Computing.," *CoRR*, vol. abs/1511.0. 2015.

[12] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," *INFOCOM, 2012 Proceedings IEEE*. pp. 945–953, 2012.