An Energy Optimizing Scheduler for Mobile Cloud Computing Environments

Manjinder Nir*, Ashraf Matrawy† and Marc St-Hilaire†
*Department of Systems and Computer Engineering
†School of Information Technology
Carleton University, Ottawa, Canada.
{manjinder.nir, ashraf.matrawy, marc_st_hilaire}@carleton.ca

***Abstract*—In mobile cloud computing, mobile devices seek to minimize computation time and/or energy consumption based on task related or user defined constraints. In earlier work [1], we proposed to minimize the total energy consumption across all the mobile devices in a cyber foraging system using a scheduler that runs in a centralized broker node, in situations where a large number of mobile devices could be expected. In this paper, we extend our earlier task scheduling problem for a large number of mobile devices to a mobile cloud computing environment. We optimally solve the task scheduling problem for task assignment to minimize the total energy consumption across the mobile devices subject to user defined constraints. Our task scheduler model at the centralized broker optimally offloads tasks and provides significant reduction in energy consumption compared to the energy consumption when tasks are offloaded from the centralized scheduler without optimization.**

Keywords*—*Mobile cloud computing; cloud computing; task scheduler; offloading.

I. Introduction

Mobile cloud computing enables mobile devices to augment their resources through computation offloading of a task to resources in a cloud. The computation offloading stems from enabling mobile devices to run heavy applications and saving their resources including battery energy [2], [3], [4]. However, computational offloading of a task involves additional data communication, which may increase the task's completion time and/or energy consumption in transferring task related data. Thus offloading is beneficial if the cost of executing a task on a cloud is less than cost of running the task locally [5], [6].

In previous work [7], Nir and Matrawy proposed resource monitoring at a centralized broker node on behalf of mobile devices in a large cyber foraging system [8], [9]. This proposed approach [7] helped in (i) reducing the time delay experienced by mobile devices due to resource monitoring of multiple surrogate nodes, and (ii) reducing the communication traffic created from a large number of mobile devices during resource monitoring. In our recent work [1], we focused on the task scheduling process by using a broker node in a large cyber foraging system. The proposed scheduler helped in minimizing the amount of energy consumed across all mobile devices in the cyber foraging system with latency constraints.

In this paper, we have modified the task scheduler model proposed in [1] such that it can be applied to mobile cloud computing environments. More precisely, we propose a mathematical model for the centralized task scheduler problem in a mobile cloud computing system. The model minimizes the total energy consumption across all mobile devices of the system. The main differences between this task scheduler model and the one in [1] are with respect to the mobile cloud computing environment. In this model, we assume that cloud providers have unlimited computing resources (virtual machine instances) in contrast to limited computing resources (surrogate nodes) in a cyber foraging system. In the previous model, we considered fixed data rate between the mobile devices and the surrogate nodes and fixed delay time tolerance for tasks of the mobile devices. However, in this paper, we consider varying delay time tolerance for every task, and this, in turn, defines the constraint for minimum required data rate for a given input/output data size of the task. Since the centralized task scheduler has a resource list of available cloud resources, it can assist mobile devices to schedule their tasks and can benefit them in avoiding resource monitoring prior to task scheduling and consequently save their time while task scheduling.

The structure of the paper is as follows. We present the related work and the problem statement in Section II. Section III introduces the proposed task scheduler model. The evaluation of the model is explained in Section IV followed by a discussion of the results in Section V. Finally, Section VI concludes the paper.

II. Related Work & Problem Statement

In a mobile device, the rationale behind computation offloading to a remote execution location is to save resources (including battery energy) on the mobile device and/or reduce the computation time of the task. Existing works related to computation offloading, consider offloading from a single mobile device to a server [2], [4], [10], [11]. We consider an area where we can expect large number of users finding cloud resources to offload their tasks such as a conference hall. For beneficial computation offloading, mobile devices have to connect with a number of cloud providers and have to decide which available cloud resources could benefit them through computation offloading. Thus whenever mobile devices have to offload, prior to offloading, they have to contact the various cloud providers. In a scenario where we expect a large number of mobile devices, repeated communication between these mobile devices and the cloud providers can cause communication overhead in the network. Also mobile devices experience delay while offloading. Thus there is a need for a centralized task scheduler which could help mobile devices in task scheduling and could benefit them with resource optimization. A hybrid cloud computing system uses centralized scheduler that optimizes private and public resource allocation with deadline and monetary constraints [12], [13]. The work in [14] provides a system for energy saving in smartphones in mobile cloud

978-1-4799-3088-3/14/$31.00 ©2014 IEEE

computing. The work in [2] minimizes energy consumption in mobile devices by optimally scheduling the transmission rate with time delay constraints. Similarly, the work in [15] does qualitative analysis to decide whether to offload or not with an aim to save energy. Existing systems on mobile cloud computing do not consider task scheduling and energy optimization for a large number of mobile devices.

We perform task scheduling at a centralized node in the system. We refer to this central node as the broker node. The resource monitoring service in the task scheduler periodically gathers the resource description about the availability of resources from the various cloud providers. When mobile devices contact the task scheduling service for task offloading, it decides the appropriate offloading location on behalf of the mobile devices by minimizing the total energy consumed across all mobile devices with data rate constraints. Multiple broker nodes, load balancing and reliability should be part of the system design in order to avoid having a single point of failure if only one broker is used. However, these issues of load balancing and reliability along with security and access authentication to cloud providers are outside the scope of this paper. In the following section, we present the proposed centralized task scheduler model.

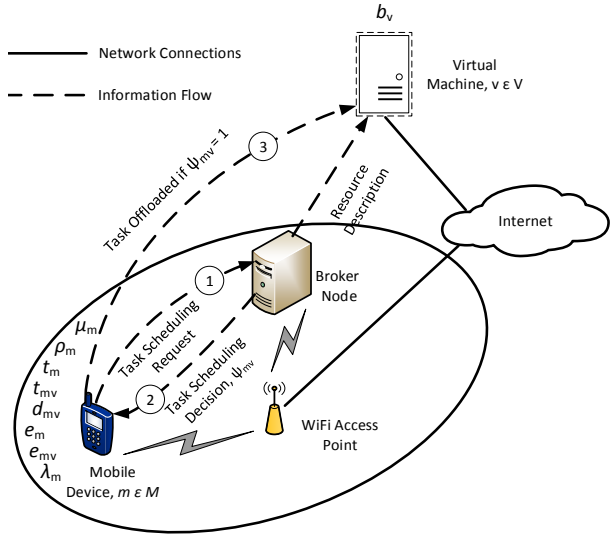

Fig. 1. Task Scheduler Model for Mobile Cloud Computing Environments.

III. Task Scheduler Model

The proposed task scheduler model finds an optimal task assignment such that the total energy consumption across all the mobile devices in a mobile cloud computing system is minimized. In the model, we set the user defined delay time tolerance parameter for every task that is being offloaded, and this, in turn, puts limit on the completion time of the tasks and defines the constraint for minimum required data rate for a given input/output data size of the tasks. The graphical illustration of the model is shown in Figure 1. The various cost and constraint parameters associated with the mobile devices and the cloud resources are shown. We refer to cloud resources as virtual machines. To this end, we propose a mathematical model based on the following assumptions and notation.

A. Assumptions

- To simplify initial analysis, we assume each mobile device has a single task to offload/execute at any given time.
- All the mobile devices and the broker node are connected to a WiFi network and all mobile devices can communicate with the broker node. The mobile devices and the broker node in the system access cloud resources through Internet connectivity via a WiFi access point (Figure 1).
- We do not consider the monetary cost of renting infrastructure resources and transferring inbound/outbound traffic to/from the cloud providers
- We assume that cloud providers have infinite infrastructure resources (virtual machine instances). Therefore none of the tasks is denied from offloading due to lack of available resources at the cloud providers.

B. Notation

The following notation is composed of sets, cost parameters, constraint parameters and the decision variable of the model.

1) Sets:

- $\mathcal{M}$, the set of all mobile devices, where mobile device $m \in \mathcal{M}$.
- $\mathcal{V}$, the set of all virtual machine instances available from cloud providers, where virtual machine instance $v \in \mathcal{V}$.
  - b_v, available data rate between all mobile devices in $\mathcal{M}$ and virtual machine instance v. All the mobile devices have the same data rate to a given virtual machine instance.

2) Cost Parameters:

- e_m, energy consumption in mobile device m when its task is executed locally.
- e_{mv}, energy consumption in mobile device m when its task is executed remotely at virtual machine instance v.

3) Constraint Parameters:

- t_m, execution time of task from mobile device m when executed locally.
- t_{mv}, execution time of task from mobile device m when executed remotely on virtual machine instance v.
- μ_m, amount of input data required from the task of mobile device m.
- ρ_m, amount of output data generated by the execution of the task from mobile device m.
- d_{mv}, delay time of the task from mobile device m when executed on instance v. The task's delay time includes the following components: t_{send}, time to send input data (μ_m) of the task; t_{mv}, time to remotely execute the task; and t_{rec}, time to receive the output data (ρ_m) from the instance. The delay time can be expressed by the following equation.

$$d_{mv} = t_{send} + t_{mv} + t_{rec} \quad (1)$$

- λ_m, delay time tolerance parameter of the task from mobile device m when executed on a remote virtual machine instance.

4) Decision Variable:

- ψ_{mv}, binary variable such that $\psi_{mv} = 1$ if and only if the task from mobile device $m \in \mathcal{M}$ is offloaded to virtual machine instance $v \in \mathcal{V}$, otherwise, $\psi_{mv} = 0$. The task runs locally if and only if

$$\sum_{v \in \mathcal{V}} \psi_{mv} = 0$$

The values of ψ_{mv} are subject to the assignment constraint explained in Equation (6). Ψ is the set of all the decision variables, where a decision variable $\psi_{mv} \in \Psi$, $m \in \mathcal{M}$ and $v \in \mathcal{V}$.

C. Cost Function

The cost function, denoted as $\mathcal{G}(\Psi)$, represents the total energy consumption of all the mobile devices whether executing their tasks locally or remotely. In Equation (2), the first term represents the total energy consumption by the tasks that are offloaded to virtual machine instances (when $\psi_{mv} = 1$). The second term is the total energy consumption by the tasks that are executed locally on the mobile devices (when $\psi_{mv} = 0$)

$$\mathcal{G}(\Psi) = \sum_{\substack{m \in \mathcal{M} \\ v \in \mathcal{V}}} e_{mv}\psi_{mv} + \sum_{\substack{m \in \mathcal{M} \\ v \in \mathcal{V}}} e_m(1 - \psi_{mv}) \tag{2}$$

D. The Model

The objective function of our Task Scheduling Problem for Cloud Computing Environment (TSPCCE) is to minimize the total energy consumption, which is equivalent to maximizing the total energy saving across all the mobile devices. The model can be given as:

TSPCCE:

$$min \quad \mathcal{G}(\Psi) \tag{3}$$

subject to:

*1) **Data Rate Constraint**:* The delay time tolerance (λ_m) parameter for a task is a user defined constraint when offloading it to a remote location. It puts a limit on the delay time (d_{mv}) of the task (Eq. (1)). The delay time constraints is given by the following inequation.

$$d_{mv} \leq \lambda_m t_m \tag{4}$$

If b_v is the available data rate between a mobile device and virtual machine instance v, then the time to send (t_{send}) the input data (μ_m) of the task and the time to receive (t_{rec}) the output data (ρ_m) generated from the execution of the task at the virtual machine can be expressed as $\frac{\mu_m}{b_v}$ and $\frac{\rho_m}{b_v}$ respectively. After solving Equations (1) and (4) for b_v, we define the data rate constraint by the following inequation.

$$b_v \geq \frac{\mu_m + \rho_m}{\lambda_m t_m - t_{mv}} \quad \forall m \in \mathcal{M}, \forall v \in \mathcal{V} \tag{5}$$

*2) **Multiple Assignment Constraint**:* This constraint imposes that a given task from mobile device m is not offloaded to multiple virtual machine instances. The assignment constraint is given by the following inequation.

$$\sum_{v \in \mathcal{V}} \psi_{mv} \leq 1 \quad \psi \in \{0, 1\}, \quad \forall m \in \mathcal{M} \tag{6}$$

IV. Evaluation

In computation offloading, generally, resource intensive tasks are considered for offloading. A task could be either CPU, memory or I/O resource intensive, or a combination of these resources. In this paper, we consider tasks which could be intensive in either of the resources, or a combination of these. For example, a task could be CPU and input data intensive, and another task could be only input and output data intensive. The type of resource intensiveness determines the energy consumption during local and remote execution. When a task is CPU or memory intensive, it accounts for the energy consumption during its local execution, while input and output data intensiveness of the task accounts for the energy consumption during offloading. We solve the TSPCCE problem with multiple data inputs, and the various input values we set are explained in the following sections.

A. Task Execution Times and I/O Data Sizes

1) Local Execution Time (t_m): The local execution time of the tasks for the mobile devices in set $\mathcal{M}$ are according to a uniform distribution $\mathcal{U}(a, b)$ within the range $a = 0$ to $b = 100$ seconds.

2) Remote Execution Time (t_{mv}): If $\mathcal{F}$ is the speed-up factor of virtual machine instance v with respect to mobile device m, then the remote execution time t_{mv} of the task is given by the following equation.

$$t_{mv} = \frac{t_m}{\mathcal{F}} \tag{7}$$

We assume that all the virtual machine instances in set $\mathcal{V}$ are four times faster than each of the mobile device in set $\mathcal{M}$ (i.e. $\mathcal{F} = 4$).

3) I/O Data Sizes (μ_m and ρ_m): The input and output data sizes of a task are set according to a Uniform distribution $\mathcal{U}(a, b)$ within the range $a = 0$ to $b = 1, 10, 20, 30, 40$ MB.

B. Delay Time Tolerance Parameter (λ_m)

The delay time tolerance parameter is a user defined parameter. It is a unit less quantity. When a task is offloaded, its value sets a limit on the completion time of the task. From (1), (4) and (7), the delay time tolerance can be given by the following inequation.

$$\lambda_m \geq \frac{1}{\mathcal{F}} + \frac{\mu_m}{b_v t_m} + \frac{\rho_m}{b_v t_m} \tag{8}$$

Consider a case when a task has negligible input (μ_m) and output (ρ_m) data size, then the value of $\lambda_m \geq \frac{1}{\mathcal{F}}$. Thus, this value is the minimum value of the delay time tolerance parameter. We set the value of the delay time tolerance parameter according to a Uniform distribution $\mathcal{U}(a, b)$ within the range $a = \frac{1}{\mathcal{F}}$ to $b = 0.3, 0.5, 1.0, 1.5, 2.0$.

C. Data Rate (b_v)

We set data rates available between mobile devices and virtual machine instances according to a normal distribution $\mathcal{N}(\mu, \sigma)$ with a mean μ of $500KB/s$ and standard deviation σ of $300KB/s$.

D. Energy Consumption

In general, energy consumption (E) is the product of the current (I) drawn across a voltage (V) for a given time (t), and is given by the relation $E = VIt$ [16]. Thus the power rating is defined as:

$$PowerRating(VI) = \frac{E \quad (\text{in J})}{t \quad (\text{in s})} \tag{9}$$

We assume that the current ratings of the WiFi radio during the sending and receiving states are $0.0857A$ and $0.0528A$ respectively [16], and the voltage rating of the mobile devices battery is $3.8V$. We assume that the power rating of the CPU in computation state is higher than the power rating of the WiFi radio in sending or receiving state, and we consider an arbitrary value for CPU power rating in computation state. The estimated power ratings of WiFi radio and CPU in their respective states are shown in Table I.

TABLE I. POWER RATINGS OF WIFI RADIO & CPU AT DIFFERENT STATES

Device	State	Power Rating (J/s)
WiFi Radio	**Send**	0.325
WiFi Radio	**Receive**	0.2
CPU	**Compute**	0.7

1) Remote Energy Consumption (e_{mv}): The remote energy consumption (e_{mv}) includes the following parameters: e_{send}, energy consumed in sending input data; e_{Idle}, energy consumed in idle state; and e_{rec}, energy consumed in receiving output data. The remote energy consumption can be expressed by the following equation.

$$e_{mv} = e_{send} + e_{Idle} + e_{rec} \tag{10}$$

Energy consumption in the idle state (e_{Idle}) of a mobile device (while waiting for the results) is negligible and can be ignored. Moreover, during this time the mobile device can do other tasks. We estimate the energy consumption in a mobile device when in sending or receiving state by the following equations.

$$e_{send} = t_{send} \times PowerRating\_send$$

$$e_{rec} = t_{rec} \times PowerRating\_receive$$

2) Local Energy Consumption (e_m): The estimation of local energy consumption when task of mobile device m is executed locally, is given by the following equation.

$$e_m = t_m \times PowerRating\_compute$$

V. RESULTS

Our TSPCCE Integer Linear Program (ILP) is implemented by using the IBM's linear programming solver called CPLEX [17]. The evaluation of the model is performed on a single server machine with Intel Xeon(R) E5420 @ 2.50GHz, quad core CPU and 8GB of RAM. The Linux OS distribution on the server is Ubuntu 12.04.2 LTS precise 64bit.

In this section, we generate problem sizes based on the number of virtual machine instances and the number of mobile devices. The number of mobile devices ($|M|$) is varied from 20, 40, 60, 80, 100, 150, 200, 250, 300 for each number of virtual machine instances. In the model, we assumed that infinite number of virtual machine instances are available; however, to generate results and to make sure that none of the mobile device is denied from offloading due to lack of resources, we vary the number of virtual machine instances from $400, 500$ and 600. To evaluate the performance of the task scheduler, we observe the energy consumption across all mobile devices in three scenarios, which are as follows.

Energy consumed without offloading: The first scenario gives us a baseline of the total energy consumption across all mobile devices $|\mathcal{M}|$ when all the tasks are processed locally. There is no broker node involved to schedule task offloading. In this case, the total energy consumption is given by Equation (2). However, since there is no offloading (i.e. $\psi_{mv} = 0, \forall m$), the total energy consumption across all the mobile devices is $\sum_{m \in M} \quad e_m$.

Energy consumed with offloading without optimization: In the second scenario, the task scheduler in the centralized broker node schedules tasks without optimizing the total energy consumption. The total energy consumption in this case is expressed by Equation (2). The values (ψ_{mv}) in the set Ψ will be obtained using one-by-one scheduling of all tasks in sequence while satisfying constraints in Equations (5) and (6). In other words, a given task will be offloaded to the first available virtual machine that satisfies all the constraints.

Energy consumed with offloading with optimization: In the third scenario, the task scheduler in the centralized broker node schedules tasks while optimizing energy consumption across all mobile devices (Equation (3)). The total energy consumption in this case is also given by Equation (2). The values (ψ_{mv}) in the set Ψ will be obtained as a result of the optimization process with the aim to minimize the total energy consumption across all the mobile devices while satisfying constraints in Equations (5) and (6).

In the second and third scenario, we observed the effect on the total energy consumption across all mobile devices $|\mathcal{M}|$ based on the following parameters; (i) number of virtual machine instances ($|\mathcal{V}|$), (ii) amount of input (μ_m) and output (ρ_m) data sizes, and (iii) delay time tolerance parameter (λ_m). In all the sections of the results, the total energy consumed across all the mobile devices ($|\mathcal{M}|$) is the average of 30 iterations. In the results, we observed that the difference in the amount of the total energy consumed in the three scenarios is very big. Therefore, we show the amount of the total energy consumed using a log scale.

A. Effect of the Number of Virtual Machine Instances

In this section, we observe the effect of (i) number of virtual machine instances, and (ii) offloading with and without optimization, on the total energy consumption. To observe the effect, the number of virtual machine instances ($|\mathcal{V}|$) is varied from $400, 500, 600$. The amount of input (μ_m) and output (ρ_m) data sizes (in MB) is set according to Uniform distribution $\mathcal{U}$ $(0, 1)$ and the delay time tolerance parameter is set according to Uniform distribution $\mathcal{U}$ $(\frac{1}{\mathcal{F}}, 1.0)$.

We assumed that each mobile device has one task to offload/execute at a given time. Moreover, one virtual machine executes one task at a given time. Thus, we set the number of virtual machines ($|\mathcal{V}|$) greater than the maximum number of mobile devices ($i.e. |\mathcal{M}| = 300$) considered in the system. We observed that the total energy consumption in the second or third approach does not vary much when the number of virtual machine instances is varied from $400, 500, 600$. Thus, in Fig. (2), we only show the results when $|\mathcal{V}| = 400$.

The results (Fig. 2) show that the total amount of energy consumed when tasks are processed locally (i.e. black line) is

always higher than the other two approaches. We observed a significant reduction in energy consumption (i.e. green and red line), when we add a centralized broker node in the architecture and employ a task scheduler which schedules task offloading on behalf of the mobile devices. We further observed the difference in energy consumption when tasks are offloaded without optimization (i.e. green line) and with optimization (i.e. red line). Our proposed task scheduler minimizes total energy consumption across all the mobile devices by carefully offloading tasks to virtual machines. For example, when $|\mathcal{M}| = 200$, if we use the scheduler without optimization the energy consumption will be $169.61J$ versus $39.17J$ when using the scheduler with optimization. This represents an improvement of 77%.

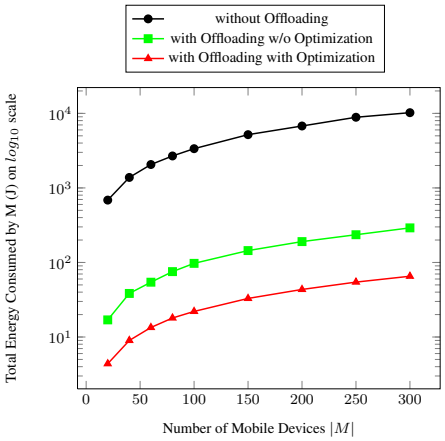

Fig. 2. Total energy consumed across all mobile devices, $|\mathcal{V}| = 400$.

B. Effect of Input & Output Data Sizes (μ_m & ρ_m)

In this section, we observe the effect of input & output data sizes on the total energy consumption across all the mobile devices. The number of virtual machine instances is set at 400 and the delay time tolerance parameter λ_m is set according to a Uniform distribution $\mathcal{U}(\frac{1}{\mathcal{F}}, 1.0)$. The input ($\mu_m$) and output ($\rho_m$) data sizes (in MB) are set according to a Uniform distribution $\mathcal{U}(0, b)$, b = $(10, 20, 30, 40)$. The results in Figures 3, 4, 5 and 6 show that the total amount of the total energy consumption across all mobile devices increases with the increase in the amount of input and output data sizes. Also with the increase in data sizes the total energy consumed is approaching towards energy consumed in non-offloading approach (i.e. black line). This observation suggests that though offloading saves computation energy consumption in mobile devices, however large data sizes increase energy consumption during data transfer.

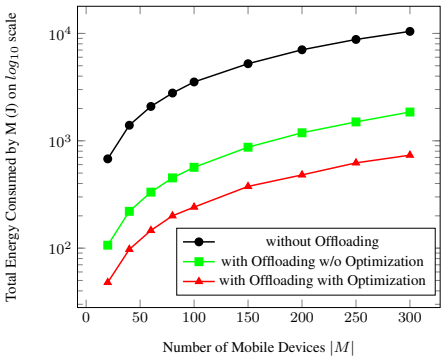

Fig. 3. Total energy consumed across all mobile devices, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 10)$.

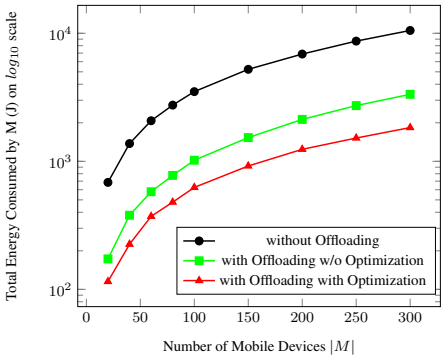

Fig. 4. Total energy consumed across all mobile devices, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 20)$.

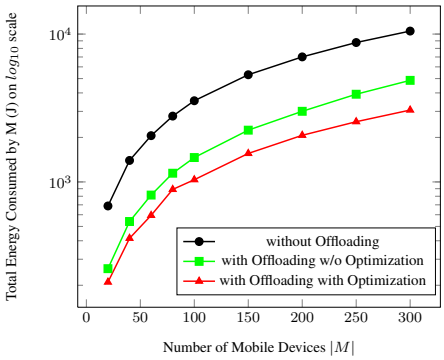

Fig. 5. Total energy consumed across all mobile devices, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 30)$.

Woohoo! You've read all the important messages in your inbo

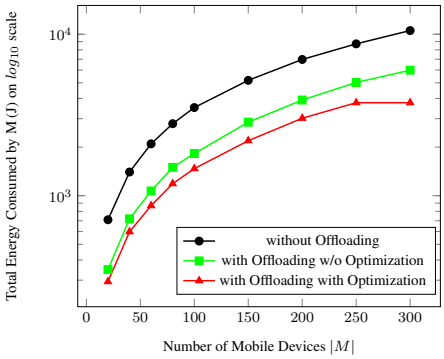

Fig. 6. Total energy consumed across all mobile devices, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 40)$.

C. Effect of Delay Time Tolerance (λ_m)

In this section, we observe the effect of the delay time tolerance on the total energy consumption across all mobile devices in the third scenario i.e offloading with optimization. The number of virtual machine instances ($|\mathcal{V}|$) is set at 400. The delay time tolerance parameter (λ_m) is set according to a Uniform distribution $\mathcal{U}(\frac{1}{\mathcal{F}}, b)$, b = $(0.3, 0.5, 1.0, 1.5, 2.0)$.

We observed the effect of the delay time constraint on the total energy consumed in relation to different input and output data sizes. In Figure 7, the input/output data sizes (in MB) are set in a range $\mathcal{U}(0, 1)$. The results show that the amount of energy consumption for various values of λ_m is similar. The reason for these results could be that the data sizes are small and for every value of λ_m the constraint in Equation (5) is satisfied and almost all tasks of all the mobile devices are offloaded.

Further to see the effect of λ_m when there are higher amounts of data sizes, we set data sizes (in MB) in a range $\mathcal{U}(0, 20)$ (Figure 8) and $\mathcal{U}(0, 30)$ (Figure 9). The results show that at higher values of data rates, the amount of energy consumption increases as the value of λ_m becomes small. The results in Figures 8 and 9 reveal that when input/output data size increases and λ_m becomes small then, while offloading, the number of tasks that do not satisfy the constraint in Equation (5) increases. Thus the number of tasks which are not offloaded increases and this leads to more energy consumption.

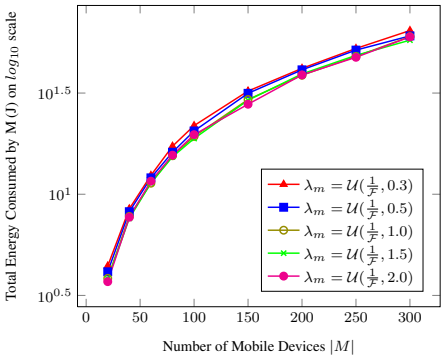

Fig. 7. The effect of delay time tolerance (λ_m) on energy consumption, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 1)$.

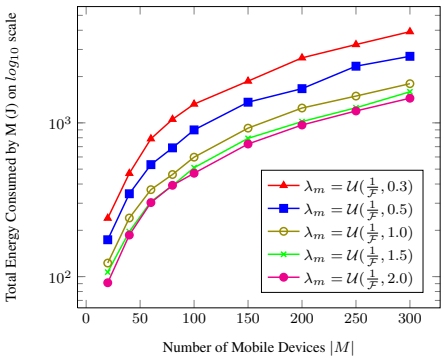

Fig. 8. The effect of delay time tolerance (λ_m) on energy consumption, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 20)$.

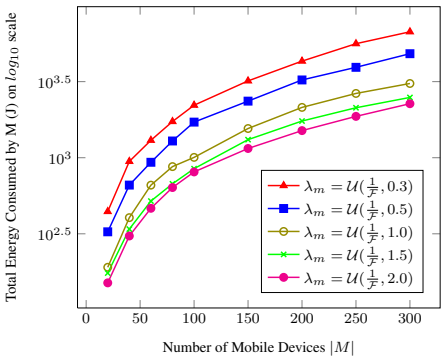

Fig. 9. The effect of delay time tolerance (λ_m) on energy consumption, when μ_m & ρ_m (in MB) are within range $\mathcal{U}(0, 30)$.

VI. Conclusion & Future Work

The model proposed in this paper provides an optimal solution for task assignment while minimizing total energy consumption across all mobile devices in a mobile cloud computing environment. This is done subject to user defined constraints. Using our proposed task scheduling model, we showed that the total energy consumption across all mobile devices is less than the total energy consumption when tasks are offloaded using centralized task scheduler without optimization. In the future, this work could be extended to model multiple tasks per mobile device and to consider task priority and future mobile networks.

Acknowledgement

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] M. Nir, A. Matrawy, and M. St-Hilaire, "Optimizing Energy Consumption in Broker Assisted Cyber Foraging Systems," in *International Conference on Advanced Information Networking and Applications (AINA-2014)*. IEEE, 2014.

[2] Y. Wen, W. Zhang, and H. Luo, "Energy-Optimal Mobile Application Execution: Taming Resource-Poor Mobile Devices with Cloud Clones," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2716–2720.

[3] 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," in *INFOCOM*. IEEE, 2012, pp. 945–953.

[4] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making Smartphones Last Longer with Code Offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[5] A. Miettinen and J. Nurminen, "Energy Efficiency of Mobile Clients in Cloud Computing," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. USENIX Association, 2010, pp. 4–4.

[6] K. Kumar and Y. Lu, "Cloud Computing for Mobile Users: Can Off. Computation Save Energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[7] M. Nir and A. Matrawy, "Centralized Management of Scalable Cyber Foraging Systems," in *Proceedings of the 4th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN)*, vol. 21. Elsevier, 2013, pp. 265–273.

[8] J. Flinn, S. Park, and M. Satyanarayanan, "Balancing Performance, Energy, and Quality in Pervasive Computing," pp. 217–226, 2002.

[9] M. Kristensen, "Scavenger: Transparent Development of Efficient Cyber Foraging Applications," in *Pervasive Computing and Communications (PerCom), International Conference on*. IEEE, 2010, pp. 217–226.

[10] B. Aggarwal, P. Chitnis, A. Dey, K. Jain, V. Navda, V. N. Padmanabhan, R. Ramjee, A. Schulman, and N. Spring, "Stratus: Energy-Efficient Mobile Communication using Cloud Support," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 477–478, 2011.

[11] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, "Cuckoo: A Computation Offloading Framework for Smartphones," in *Mobile Computing, Applications, and Services*. Springer, 2012, pp. 59–79.

[12] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-Optimal Scheduling in Hybrid Iaas Clouds for Deadline Constrained Workloads," in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*. IEEE, 2010, pp. 228–235.

[13] R. Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-Efficient Scheduling Heuristics for Deadline Constrained Workloads on Hybrid Clouds," in *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*. IEEE, 2011, pp. 320–327.

[14] F. Xia, F. Ding, J. Li, X. Kong, L. T. Yang, and J. Ma, "Phone2cloud: Exploiting Computation Offloading for Energy Saving on Smartphones in Mobile Cloud Computing," *Information Sys. Frontiers*, pp. 1–17, 2013.

[15] H. Wu, Q. Wang, and K. Wolter, "Tradeoff Between Performance Improvement and Energy Saving in Mobile Cloud Offloading Systems," *Internatioanl Conference on Communications: Ist International workshop on Mobile Cloud Networking Services (MCN)*, pp. 738–742, 2013.

[16] H. Wu, S. Nabar, and R. Poovendran, "An Energy Framework for the Network Simulator 3 (ns3)," in *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, 2011, pp. 222–230.

[17] "Cplex: IBM's Linear Programming Solver," http://www.ilog.com/product/cplex/.