Services Transactions on Cloud Computing (ISSN 2326-7550) Vol. 3, No. 1, January-March 2015

CELLCLOUD: TOWARDS A COST EFFECTIVE FORMATION OF MOBILE CLOUD BASED ON BIDDING INCENTIVE

Shahid A. Noor, Ragib Hasan, Md Haque
Department of Computer and Information Sciences
University of Alabama at Birmingham
{shaahid, ragib, mhaque}@cis.uab.edu

Abstract

In recent years, cloud computing has become one of the most dominant computing paradigms. Researchers have explored the possibility of building clouds out of loosely associated mobile computing devices. However, most such efforts failed due to the lack of a proper incentive model for the mobile device owners. In this paper, we propose CellCloud — a practical mobile cloud architecture, which can be easily deployed on existing cellular phone network infrastructures. CellCloud is based on a novel reputation-based economic incentive model in order to compensate the mobile device owners for the use of their phones as cloud computing nodes. CellCloud offers a practical model for performing cloud operations, with lower costs compared to a traditional cloud. We provide an elaborate analysis of the model with security and economic incentives as the major focus. Along with presenting a cost equation model, we perform extensive simulations to evaluate the performance and also analyze the feasibility of our proposed model. Our simulation results show that CellCloud creates a win-win scenario for all three stakeholders (client, cloud provider, and mobile device owners) to ensure the formation of a successful mobile cloud architecture.

Keywords: mobile cloud; bidding; challenges; trustworthiness; cost model;

1. INTRODUCTION

Cloud computing is a well-known computing model thanks to its competitive price, performance, and expandability. However, there is a tradeoff associated with cloud computing — hidden costs make it infeasible compared to private hosting [1], [2]. From the operational and structural point of view, the fixed structure of cloud data centers can cause underutilization of resources if there is a rapid decrease in clients' demands for cloud services. Recently, chief providers of cloud service like Amazon and Microsoft did not succeed in earning the revenues they initially expected to get, because of the unforeseen shutdown of the government budget [3]. One of the main reasons for this deficit in the expected revenue is due to the inflexibility of these organizations to contract and expand the resources based on the client requirements. We argue that, a cloud service can be designed that is not subject to underutilization of resources, if the servers themselves could be outsourced from the cloud service providers to individuals with excess resources. In this model, using mobile devices, it is possible to form a highly scalable ad hoc mobile cloud with low infrastructure set up cost and time.

Using mobile devices, it is possible to create a highly scalable, ad hoc mobile cloud with an easy to avail infrastructure, low budget, and short time frame. For this reason, mobile cloud computing is introduced where unlike a traditional cloud, a virtualized interface is formed using mobile devices.

Researchers have defined mobile cloud from two aspects. According to the first aspect, mobile cloud computing is an infrastructure where mobile users use backend cloud system for storing and processing data required to run an application [4]. The second aspect contends that, mobile cloud computing enhances the storage and computational power of the cloud system by using the unused resources of mobile devices [5]. In this paper, we would like to shed light on the second aspect of mobile cloud. A few applications utilize mobile sensed data, which is both lengthy and costly in sending to the traditional cloud for processing. A better approach would be to process data on a local basis using the mobile cloud as explained in the second aspect. Another benefit of the second aspect of mobile cloud is the accessibility of millions of unused mobile devices. The survey of Lockout Inc. data show that around 20, 16, and 19 percent of the people have respectively one, two, and more than two unused mobile devices [6]. We argue that the computing capability of such devices can and should be utilized.

There are many benefits of choosing a mobile cloud over traditional cloud. The *first* benefit is that a mobile cloud requires low set up and maintenance costs compared to traditional cloud. The *second* benefit is that a mobile cloud can be expanded to suitably keep pace of the growing demands of a client. A *third* benefit is that tasks can be effortlessly distributed and transferred among mobile devices as needed since the infrastructure of mobile network is already available in most places. Finally, according to Alzain et al. [7], Bendahmane et al. [8], and Lagar et al. [9], one can summarize that maintaining higher redundancy in task computation guarantees more accurate results. For this reason, there is a higher possibility of obtaining legitimate results in a mobile cloud than a traditional cloud since there are more unutilized mobile devices that can be used for the repetitive computation of a single task.

Most of the research performed up to now focus on the first aspect of mobile clouds which utilizes the cloud in the backend to enhance the storage and computational power, battery longevity, safety, and security of mobile device [10], [11], [12], [13]. Most researchers have not foreseen a more innovative approach where the use of mobile devices plays an integrated part of a cloud [14], [15]. Low storage and computing power of the unused mobile devices were the biggest stumbling blocks for researchers to exploit the opportunity of forming mobile clouds with these devices. However, with the advance of technology, researchers have just started considering the second aspect of mobile clouds where mobile devices are used as integral part of a cloud [14], [15]. Though these models [14], [15] include architecture for forming a mobile cloud, the absence of appropriate cost/incentive model fails to motivate the mobile device owners to participate. These models also do not address the feasibility of utilizing these unused mobile devices from the provider's perspective. To solve this, we introduce **CellCloud** – a practical mobile cloud architecture, which can be easily deployed on existing cellular phone network infrastructures. In CellCloud, we address these issues by exploring a bidding strategy for providing incentives to mobile device owners, and also quantify the benefits achieved by cloud providers by using mobile devices as cloud nodes.

In CellCloud, mobile devices known as *bidders* are hired following a bidding process. During the bidding process, each bidder is offered monetary incentive based on their available resource and rating point. The rating point determines the trustworthiness of the bidder for a particular task. Based on client requirement, CellCloud provider hires the required number of bidders for a task, divides the task into smaller subtasks, and distributes them among bidders for computation. The CellCloud provider uses a MapReduce based scheme for its computations. The overall model proves to be cost effective for both cloud providers and mobile owners creating a win-win situation for all the stakeholders. Initial results from CellCloud were presented in [16]. In this paper, we provide a comprehensive discussion of our expanded model, and present extensive and new simulation results from additional benchmarks of various aspects of CellCloud's performance in order to demonstrate the feasibility of this model.

Contributions:

1) To the best of our knowledge, CellCloud is the very first attempt to form a mobile cloud using the network of the mobile operators and unused mobile phones. We introduce a bidding process for forming the mobile cloud where mobile device owners can submit their free resources during bidding and get incentives for their resources.

2) We introduce the novel concept of rating points associated with the bidders (mobile devices) to ensure trustworthiness and reliability of the service in hand. Moreover, our model enables users to choose different service costs based on the rating level of the computing nodes, thus providing a unique opportunity to the clients.

3) We provide simulation results on an LTE network topology using the NS-3 simulator to demonstrate the feasibility of the CellCloud model.

The rest of the paper is organized as follows. Section 2 describes the motivation of our research work. In section 3, we discuss some of the related works on mobile cloud. Section 4 introduces our mobile cloud architecture. We describe the strategies for assigning rating points, measuring the cost of the operation, and selecting base stations and bidders in section 5, 6, 7, and 8 respectively. Section 9 elaborates possible challenges in mobile cloud. We define some policy for the client and the bidder in section 10 followed by experimental results in section 11. Some possible applications of CellCloud are mentioned in section 12. Finally, we conclude with discussion and future directions in sections 13 and 14 respectively.

2. MOTIVATION

Mobile cloud computing requires a significantly smaller amount of initial set up cost as compared to a traditional cloud system. The primary reason of this low set up cost is that, it utilizes the unused mobile devices to create a cloud platform. A study by Lockout found that, approximately 52%mobile users intend to donate their old sets for charitable use [6]. Moreover, it is also observed that, mobile devices are in idle state for 89% of the time in a day and during that time the devices use less than 11% of total CPU power [17]. Therefore, with no obvious harms and associated economic benefit, we believe that it would be very easy to motivate the mobile device owners to share their unused mobile devices on a cloud platform.

Scalability of the client tasks is another big reason to choose mobile cloud than traditional cloud systems. Sometimes, there can be more demand for resources than expected. In general, a good amount of resources remain unused in a public cloud, as the providers want to be on the safe side. Therefore, if at any time there is a certain decrease in client demand, some resources will be unused. Recently, Amazon, Microsoft, and some other cloud services failed to earn expected revenues due to the unexpected shutdown of government budget due to the stalemate in the US legislature [3]. On the other hand, in our CellCloud architecture, we always hire bidders on an on-demand basis. Since the infrastructure of the mobile phone network is already deployed and available, the cost for keeping continuous connection between the bidders and the base stations is almost negligible. We can always have a good number of reserve bidders. Therefore, our CellCloud architecture will be able to handle both the sudden increase and decrease of clients without any financial downfall.

A mobile cloud can reduce the computational cost significantly as compared to the traditional backend cloud system. Mahesri and Vardhan showed that the average power consumption of a personal computer with a 1.3 GHz processor in idle state is 13.13W [18] whereas, that of a mobile phone with a 400 MHz processor is only 268.8mW [19]. Hence, the total power consumption in a mobile cloud is much less than a traditional cloud. Therefore, we can include more bidders in our CellCloud to achieve the same computing capabilities as provided by a traditional cloud. This comparatively higher power consumption issue also plays a big role in higher operational cost for traditional cloud services.

Ensuring trustworthiness during computation is another major reason to choose a mobile cloud over the traditional clouds. Bendahmane et al. discussed two popular methods for ensuring the authenticity in cloud computation: majority based voting and m-first voting system [8]. However, both of these methods use multiple virtual devices for computing a single task. We use a similar approach in our CellCloud for enhancing trustworthiness. Since the level of trust is associated with the level of redundancy in task computation, availability of larger number of free bidders will help us in getting more precise results. In CellCloud, we always have a large number of unused bidders, which can be used for redundant computation.

3. Related Work

Mobile computing is used for developing several applications in the field of distributed computing. Beberg et al. proposed Folding@home, a distributed computing methodology, for reproducing biophysical methods [20]. As indicated by their building design, volunteer users with an interest to participate in the computation allow their personal computers to be used. The server sends the users the data about a specific work unit, which is the mixture of some input files needed for completing a job inside a certain time period. A header is connected within a work unit to set the core type while the version inside a work unit is used to download and process the core. A core can be conceived as an executable file, which contains input files and generates output. The output created by the designated computational core is sent to the server. Since cores are autonomous from customer in this manner, this methodology can do any kind of processing and the overhaul of centers does not require reinstallation of any software. The significant downside of this methodology is that once a task is given, client has to follow and download the essential core type manually. Therefore, it is unlikely that they would like the complexity associated with the operation. In contrary, in our CellCloud architecture, the inputs are provided along with the requirements during the time a bidder participates on bidding. Based on the instruction provided by cloud, the bidder will just perform computation.

Condor is a very famous project that is initiated by the researchers of the University of Wisconsin Madison [21]. It manages, circulates, and maintains a wide scope of computing systems. The project has the flexibility to match any input request for resource, assigning checkpoint along with migration facility. Moreover, it allows remote system calls to run jobs remotely. The jobs with higher priority are executed first without any interruption. The lower priority jobs are executed while the CPU is in idle and transferred to another one as soon as the CPU becomes busy. The user job is taken by the agent associated with the Condor kernel, while the resources are monitored by the matchmaker. As soon as a match is found, the job is transferred to that specific resource. One major problem of Condor project is that when a higher priority job comes and the system does not have enough resource, the lower priority jobs are interrupted. Hence, estimating the task completion time is very difficult for the client. On the contrary, jobs are independent of each other in our CellCloud architecture. Resources are allocated for a client as soon as a job is accepted. If there is not enough resource for satisfying the client requirements, then CellCloud will not accept the task. Therefore, it is easier for client to anticipate the time and cost of task completion in CellCloud.

Miluzzo et al. proposed the concept of mClouds where group of mobile devices, known as mDevs, are brought together to form a cloud computing platform [15]. Whenever a mobile device wishes to compute a task, which requires larger resources than it currently has, the device broadcasts a solicitation message to inform the other mobile devices to join its mCloud formation. The mobile device divides and distributes the task among the mCloud devices. If the task is too large to finish even after forming the mCloud, then the device uses the backend cloud system for the remaining subtasks, as no mDevs are available. In mCloud system, mobile users have to decide when to form the mCloud and whether joining on mCloud will be beneficial or not. The primary problem in mCloud system is to ensure security since anyone having a mobile device can join without verification. Therefore, the presence of a rogue device can lead to wrong output. On the other hand, all the devices used in our CellCloud are verified by the operator providing cloud service before they are included on cloud. Therefore, we can ensure more accurate and trustworthy results in CellCloud than the mCloud system.

Researchers from the Space Science Laboratory of University of California created the SETI@home project to explore the presence of life in the universe [22]. In their project, they consolidate immense computing power distributed all around the world to examine radio telescope signals come from space. A large number of data are broken into smaller chunks and distributed among a large number of computers for processing. Result obtained from each computer is organized by central repository. Compared to the typical distributed systems, SETI@home uses more variety of resources distributed among diverse locations. However, the major problem in SETI@home project is that the tasks are distributed only if the resources are in idle state. Therefore, it rarely provides any real time solution for a task. Some tasks might need to be finished within a specific deadline. On the other hand, in our CellCloud architecture, client can inform the deadline for their task and CellCloud selects the resources based on the client requirements.

Dashti et al. introduces an approach to use smartphone sensors for effectively and efficiently detecting the effect of an earthquake [23]. In their approach, smart phones are placed into an artificially shaking table. The shaking table can move in every direction and can horizontally accelerate up to 1.5 g even to a 100,000 lb object. The time needed for the smartphones to send their data, along with the position and orientation, is determined and used during prediction. A shaking meter continuously detects shakes and only starts recording while the meter drops below a threshold. The emergency responders can summarize the result sent by those sensors and take the necessary steps based on the summary after an earthquake. Their approach does not produce sparse result unlike the result produced by the existing methods. Direct human observation during the earthquake might help in detecting the actual situation of earthquake but it is highly related to how fast the user responses. Moreover, untrained users might provide misleading information. Therefore, using smartphone sensors for observing real time earthquake data as proposed in their research work will be both inexpensive and efficient as they are now easily available and the sensors perform quite well in sensing the environment.

Mednis et al. discuss the uses of mobile sensors to detect the real time irregularity on roads [24]. Based on several parameters such as, average load on road, vibration on road, the possibility of a road being damaged at a specific time is determined. The usage of ground penetration radar or commercial products for surface analysis is economically infeasible. On the other hand, collecting photos to determine damage and hazard on the road requires complex image analysis along with human interaction. Therefore, mobile sensors can be both automated and economical solution for determining the road hazards. Four different approaches are used for detecting pothole on the road. The first approach Z-THRESH compares the accelerometer reading values with a threshold value and any value exceeding the threshold indicates the possibilities of pothole. In second method, the difference between two values above a threshold is used to determine the pothole. The third method first determines the standard deviation of the acceleration on vertical X-axis, which is compared with the threshold to detect pothole. The fourth method considers threshold values in the entire three axes for measuring pothole. One problem on their proposed work is, there can be several values above a threshold. But it is unclear which two values will be taken from those values. Similarly the authors never defined clearly the term sliding window. So its hard to imagine the effect of the size of sliding window on determining true positives and true hit. From their experimental result it is also seen that the performance of all the methods other than Z-DIFF is not convincing for detecting drain pits.

Angin et al. proposed a Context-aware blind navigation system, which uses the resources of cloud for complex computations [25]. In their method, a cameral is attached with the sunglass, which captures the visual records and sends it to the mobile device via Bluetooth. The mobile device senses the voices and its integrated camera determines the position. The mobile device processes the data for identifying the context. The positioning and direction is precisely determined by using the combination of GPS, Wi-Fi access point, and cell tower triangulation along with a compass. The authors used android as a mobile platform and Amazon EC2 as a cloud platform. For detecting the 3D real-time images, they choose time-of-flight range cameras. Their proposed method, in addition to reaching the destination properly, also supports blind people to effectively track their personal belongings along with the identifying people surrounded by them. In addition, it provides several advantages as compared to the normal stereo cameras such as, better detection of actual dense, simpler and efficient processing etc. Moreover it can guide during outdoor navigation such as how to cross an intersection in urbane area, detecting an obstacle, locating a bus stop along with the pedestrian signal.

Hoang et al. proposed a mobile cloud architecture that composed of sensor and mobile agent component which are responsibility to collect and manage data from the environment [26]. They included several features to maximize the utilization ratio. For example their asynchronous message mode (AMM) activates a mobile application, which sends a request to cloud server and wait until a response comes from the server. For implementing AMM they uses a push mechanism, which helps to run application in the background so that the energy can be saved. A database is embedded with their system, which relocates the relevant data when the connection is lost. The context aware mobile and middleware section of their proposed architecture used the contextual information such as quality of session, condition of network, temperature, humidity location, etc. during processing the information. They use an intelligent learning model for creating and updating their context repositories. This contextual form of data can be used to reduce the energy consumption of a specified application. A middleware layer is placed between data acquisition layer and back end cloud layer to handle the power of the context aware data along with the activation of emergency if required and dynamic allocation of the back end cloud resources. However, It is not clear from their work that how does they handle the power or network failure. No algorithm is mentioned for handling such unwanted situations.

All the aforementioned approaches have some common problems such as, the motivation to participate in cloud architecture on part of the device owners is overlooked, the requirements of client for a task is not mentioned, and no cost benefit analysis is provided for client to take decision on joining cloud system. On the contrary, in CellCloud, the cloud service provider hires bidders by providing incentives. As a result, we can expect to get a large number of bidders for tasks. Clients can also submit their requirements for a task and the CellCloud provider provides the estimated cost of that task. Therefore, clients can analyze their benefits before giving any task to CellCloud.

Moreover, CellCloud provides a pricing chart for the client from where they can get the idea of the possible time and cost for their task completion. Clients can easily verify whether assigning a task to our mobile cloud will be beneficial for them or not.

4. CELLCLOUD ARCHITECTURE

The CellCloud architecture consists of a cloud central system (CCS), which is the central part of the operator cloud system. During the cloud set up, CCS sends a message to all of its base stations to inform the mobile users under their coverage area to initiate bidding. CCS determines the price for hiring a bidder based on the rating point of the bidder. The interested bidders submit the information regarding their available resources to the corresponding base stations. Each base station maintains a table consisting of the bidder id, rating point, and available resources. The high level architecture of CellCloud is shown in figure 1. However, the following two scenarios can be possible while a bidder wants to participate in a task:

Scenario 1: If the bidder is not registered and sends a request to the corresponding base station for the first time, the base station adds a new entry into its bidder information table. The base station requests CCS to assign an id for the new bidder. The id is stored into the id field of the table. A default initial rating point of 0.5 is assigned for the bidder. The bidder receives its id from the base station and stores it along with the id of the current base station.

Scenario 2: If an existing bidder requests to the corresponding base station for the participation of a task, it needs to send its id along with the id of last base station it visited for a task computation. The base station collects the information of that bidder from that last visited base station. The current base station updates its bidder table while the previous base station deletes the entry associated with that bidder.

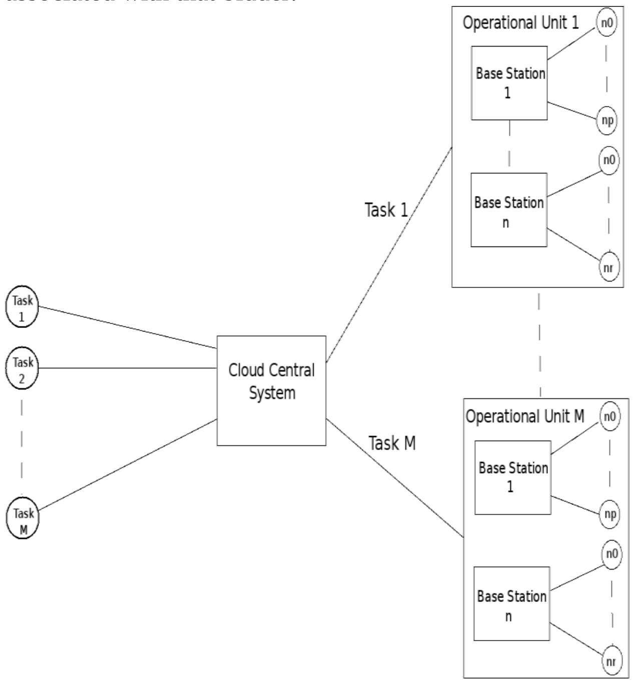

Figure 1. CellCloud Architecture

Clients are individual users who want to take cloud facility for their tasks. Clients specify their requirements for the task. The requirements include the task completion time along with the assurance that the task will be finished within the deadline. We refer this level of assurance as the reliability. Bidders with higher rating point ensure higher probability to finish the task within the given deadline. Before accepting any task, CCS verifies whether the CellCloud has sufficient resources to complete that task while maintaining the desired level of reliability. CCS contacts the base stations to submit the total free resources they can provide keeping the desired level of average rating point. Based on the information, CCS selects some base stations for that task. We refer each group of base stations for a specific task as an operational unit. Depending upon the free resources at each base station under the operational unit, CCS divides the task into several un-uniform subtasks. The base station with higher amount of free resources receives a larger subtask compared to a base station with lower amount of available resources. The operational unit divides the subtask again into several un-uniform subtasks and distributes among the bidders based on their resources. The process of mapping a task inside an operational unit is depicted in figure 2.

Each bidder sends its result to the reducer after computation. Each reducer continuously collects results from the bidders and performs reducing operation on them. Once the reducer computes the final result after reducing all the results obtained from the bidders, it sends the result to CCS. CCS collects the reduced result from each base station inside an operational unit and starts reducing the result. The final result is sent to the client. The process of reducing the result is shown in figure 3.

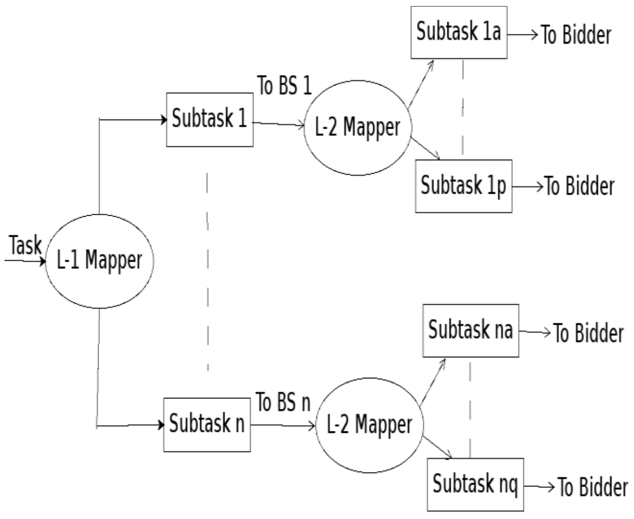

Figure 2. Process of Mapping a Task Inside Operational Unit

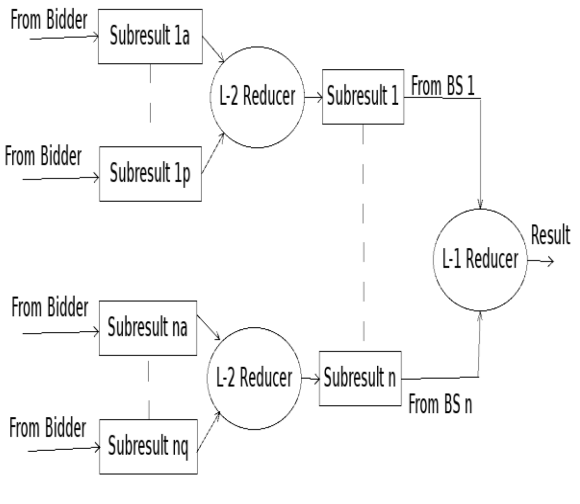

Figure 3. Process of Reducing Result Inside Operational Unit

Once a subtask is finished, base station measures the performance of each bidder under its coverage area. Based on the performance, the base station re-evaluates and updates the rating point for each bidder.

5. RATING POINT CALCULATION

The rating point denotes the level of trust the cloud provider has on a bidder of the cloud system. The rating point of each bidder ranges from 0 to 1. A bidder with a higher rating point is considered to be more trustable compared to a lower rating point bidder. Initially each bidder is assigned a rating point of 0.5. Upon the successful completion of a task, we provide a reward that will increase bidder's rating point. On the other hand, we will penalize if the bidder fails to finish the task within deadline. However, the rate of penalize will be more as compared to the rate of reward. Before sending a task to a bidder, base station will estimate the possible task completion time t_0 based on its resources. In addition, base station will define two more time. The max time t_u and the min time t_l which are 10 percent larger and smaller respectively than the original estimated time. We assign a reward of 1, 0.7 and 0.5 for completing the task within the time t_u, t_o and t_l respectively. On the other hand, a bidder will not be given any reward if it does not finish the task before t_l. The new rating point will be the average of the current rating point and the reward point. In other word, we always emphasize the most recent task during the calculation of the new rating point. Thus, if the current rating point of the users is P_c and the reward for the most recent task is P_r, then the new rating point will be,

$$P_n = \frac{P_c+P_r}{2}$$

6. MEASUREMENT OF COST AND TIME

For calculating the cost, we consider the cost associated with hiring the bidders and the cost associated with using network devices. The hiring time of a bidder is the addition of the time taken by the bidder to receive the task; the time bidder is used for task computation, and the time taken by the bidder to send the result. However, the consumed power in each mobile device during sending and receiving time is proportional to the rate at which the mobile device transfers and receives data. Suppose the power drop for sending and receiving at a rate of 1 unit/s is P_x and P_y respectively. If the bidder sends and receives data at a rate of x unit/s and y unit/s then the power drop for sending and receiving is $P_x x$ and $P_y y$ respectively. If the bidder sends data for t_s times and receives data for t_r times, then power consumption for receiving a task and sending result will be,

$$P_{sr}=t_s P_x x + t_r P_y y$$

Let us assume that the power drop during task computation is P_t in each unit of time. If the bidder takes t_t

times for computing a task then the consumed power during computing a task will be,

$$P_{task} = t_t P_t$$

Suppose the cost for consuming each unit of power is C_u thus the cost for total power consumption will be,

$$C_p = Cu\ (P_{sr} + P_{task})$$

Let us assume that bidder with rating point 0.1 receives m% profit of total cost for power consumption.

If the rating point of the bidder is R_b then the cost for hiring a bidder will be,

$$C_h = (100+m) * C_p * R_b$$

For calculating the cost associated with the use of network devices, we need to know the cost of mobile operator for transferring each unit of data within the network. Let us assume that the cost of mobile operator is C_{dt} for transferring each unit of data. Therefore, if the bidder receives p unit and sends q unit of data the cost of the operator will be $C_{dt}(p+q)$. If we assume that the mobile operator makes r% profit on its total cost for sending and receiving data then the cost associated with network will be,

$$C_n = C_{dt}(p+q)*(100+r)/100$$

Hence, the cost for completing a t unit of task by a single bidder will be,

$$C_b = C_n + C_h$$

If n bidders B_1 , B_2 , ...B_n are hired for a task with the cost of C_1, C_2, ...C_n respectively then the total cost for task completion will be,

$$C = C_1 + C_2 + \ldots + C_n$$

For each base station, we consider the following times during a task computation:

- Time to divide the task among n segments (t_d)
- Time to send the segments of a task to bidders (t_s)
- Maximum task completion time among all bidders (t_{max})
- Time to receive the results by base station from the bidders (t_r)
- Time to reduce the results (t_{rd})

Suppose n number of bidders B_1, B_2, ...B_n participate on computation where each bidder takes x_i size of task as input and produces y_i size of result. If the data transfer rate between the bidder and the base station is BW then the time that will be taken by each base station for computing its assigned task will be,

$$t_{bs} = t_d + \frac{\sum_{i=1}^{n}(x_i + y_i)}{BW} + t_{max} + t_{rd}$$

Suppose there are n base stations b_{s1}, b_{s2}, ..., b_{sn} inside an operational unit and each take t_{bs1}, t_{bs2}, ...t_{bsn} time to finish the task. If the CCS takes t_{div} and t_{red} time to distribute the task and reducing the result then the total time for a task computation will be,

$$T = \text{maxtime}(t_{bsi}) + t_{div} + t_{red}$$

Here maxtime is the maximum task completion time among all the base stations.

7. BASE STATION SELECTION STRATEGY

In the Base Station bidding strategy, the reliability scales from 0 to 1. The client submits his required reliability R along with the deadline T. CCS broadcasts a message to all the base stations and ask for total free resources it can provide with an average rating point of R. Suppose p out of n base stations reply with the resource amount r_1, r_2, ...r_p respectively. Based on the task size and the deadline, CCS estimates total resources required for finishing the tasks. CCS divides and distributes the task into several smaller subtasks in such a way that each of these p base stations receives equal load on their available resources. If M is the total resources required for finishing the task of size S and CCS selects $r_1^{/}$, $r_2^{/}$, , ... $r_p^{/}$ resources from base stations b_1, b_2, ...b_p respectively then,

$$r_i^{/} = M * \frac{r_i}{\sum_{j=1}^{p} r_j}$$

8. BIDDER SELECTION STRATEGY

Suppose a base station receives a request for resource $M^{/}$ from CCS with an average rating point of R. The base station uses the procedure as described in Algorithm 1, to select the bidders from its available bidder list.

Algorithm 1 Bidder Selection

```
1: set selectedbidders=null
2: sort bidders into ascending order $b_1, b_2, \ldots b_m$ based on the rating point $p_1$ , $p_2$ , ...$p_m$
3: find k such that $p_i >= P\ \forall i >= k$ and $p_i < P$ otherwise
4: set upperpointer=k and lowerpointer=k-1
5: push $b_{upper\ pointer}$ into selectedbidders
6: set upperpointer=upperpointer+1
7: find resourcesum which is the sum of the resources of selectedbidders
8: if resourcesum $\geqslant M^{/}$ then
9:     return selectedbidders
10: else
11:   if average rating point of selected bidders $\geqslant$ target rating point then
12:            push $b_{lower\ pointer}$ into selectedbidders
13:            set lowerpointer=lowerpointer-1
14:   else
15:            push $b_{upper\ pointer}$ into selectedbidders
16:            set upperpointer=upperpointer+1
17:   end if
18:   go to step 7
19: end if
```

Base station divides the bidders into two parts. Those bidders who have higher rating point than the target rating point R is placed into the upper part while the rest are placed into the lower part. Base station selects one bidder at a time either from upper or lower part depending upon whether the average rating point of currently selected bidders are less than or greater than the target rating point. Base station repeats the process of selecting birders until the required amount of resources for the task is achieved.

9. CHALLENGES IN BIDDING STRATEGY

We listed several unwanted circumstances in our proposed model, which are described below.

9.1 FINISHING TASK ON DEADLINE.

One of the major challenges in our architecture is to ensure that, each task will be finished before its deadline. In our CellCloud architecture, CCS only accepts a task if the cloud has sufficient resources. Task completion may be delayed due to some unwanted circumstances such as, network failure or bidder negligence. However, the bidder knows that it will receive more incentives if it has higher rating point and only finishing the task on deadline could enhance its rating point. Therefore, the bidder will sincerely try to complete the task on time.

9.2 INSUFFICIENT BATTERY POWER.

For finishing the tasks on time, mobile devices need to remain switched on. Unexpected power failure can stop computational process or result in loss of computed data. Therefore, it is very important for CCS to know the current power status of the mobile device so that it can collect the result so far computed in case of possible power failure and distributes the remaining tasks to the backup bidders. We can use the agent SystemSens in each computational device to monitor battery power [27]. A threshold level P_{th} is defined and if the power goes below that threshold then the agent communicates with the CCS to inform possible power loss. Depending on the user's requirement, the base station may consider some of the bidders as reversed bidders and may contact them when required. Base station forwards the incomplete tasks to one of those reserved bidders on the cloud system. Base station sends a confirmation to both the current bidder and the new bidder when the incomplete tasks are transferred to the new bidder.

9.3 TRUSTED COMPUTING.

Another major concern for the provider is to ensure that any malicious program inside a mobile device cannot manipulate computation or compromise data inside the mobile device. Using a trusted cloud computing platform (TCCP), we can prohibit malware to access input and output data, or stop interfering during computation [28]. To establish a TCCP, a trusted virtual machine monitor (TVMM) [29] is installed in a mobile device if the platform inside the mobile device satisfies the specification defined by the trusted computing group (TCG). TVMM prohibits even the privileged user from observing or altering the data during computation. A trusted coordinator (TC) inside TCCP certifies a platform if it finds the platform secure for computation. A bidder only accepts input data and performs computation if TC certifies it.

10. POLICIES IN CELLCLOUD

The list of policies in our mobile cloud architecture includes:

10.1 TASK SELECTION POLICY.

We expect a large number of requests from client at a certain period of time. The clients will be served on first come first serve basis. A client will be served only if the cloud has enough resources to fulfill clients' requirements. Otherwise, clients will be informed regarding the limitation of resources. The client might change its requirements and request for cloud service again. If two client come at the same time then the client which can be served with relatively less time will be served first so that more number of clients get the opportunity to use CellCloud.

10.2 WAITING TIME POLICY.

It is expected that each bidder will finish its task before the timeline defined by base station. However, there are still possibilities that a bidder fails to finish the task before deadline. Also, Bidders may loose the connection with base stations and therefore, fail to submit the result. Bidders can also refuse to do the task. Regardless of whether any bidder is working or not, mobile cloud always needs to ensure the client that their task will be finished within the deadline. We use the following steps for handling such unwanted situations:

Step-1: We include redundancy during task computation. The level of redundancy is proportional to the level of reliability client demands.

Step-2: Base station only considers the bidders who send their result before the original estimated time t_o.

Step-3: If no bidder sends its result before the original estimated time t_o, base station will still wait for receiving a result from a bidder until the min time t_m reaches.

Step-4: If all the redundant bidders fail to submit the result within the min time t_m, base station finds alternate bidder and give the task to those bidders.

10.3 PRICE AND DELAY POLICY.

We assume that mobile cloud will make 100% profits on the cost of computation. If the cost associated with computing a task is C then the price for the client will be 2C. In case of failure to finish the task within the deadline specified by the client, the mobile cloud will make 50% less profit than the profit made for successful completion of the clients' requirements.

11. EVALUATION

In this section, we present our implementation and evaluation results.

11.1 EXPERIMENTAL SETUP.

In our experiment, we used three types of mobile devices as our bidders. The first one was a Google nexus 4 with quad-core Krait clocked at 1.5GHz processor along with 2GB of RAM, the second one was a Samsung Galaxy S Plus with a 1.4 GHz Scorpion processor and 512 MB RAM. The third one was a Samsung Galaxy S4 with 1.2 GHz quad-core processor and 2 GB RAM. These mobile devices have internal or external SD card where the task was stored before executing. Similarly, the result of the computation was also stored in the SD card before sending back to the corresponding base station. We solved all the benchmark problems for a input file of size 1MB to 10MB with 1MB increment on these mobile devices and measured the time and power consumption. We calculated the average task completion time and power consumption on each of those mobile devices for a task of 1MB and considered that task as our base task. However, the sorting problem required much longer time on mobile devices due to their limited resources. Therefore, we solved the sorting problem for a input file of size 1KB to 10KB with 1KB increment and measured the time and power consumption for those tasks. We calculated the average task completion time and power consumption on each of those mobile devices for a task of 1KB and considered that task as our base task. We repeat the whole steps for 10 times and measure the task completion time and the power consumption. From the sample run, we found that the task completion time on each type of mobile device is almost proportional to the task size for specific types of task. Hence, we simplified our model by comparing any task size with the base task size. For example, if the ratio between the current task size and base task size is S_r and the base task completion time is T_b, then the current task completion time is S_rT_b. The task completion time varies depending on the types of mobile devices. Therefore, instead of considering a fixed base task completion time, we considered a random time within a defined range as base task completion time for each bidder in our simulation. Similarly, we measured the average power drop in each second while solving the three-benchmark problems.

As part of the evaluation, we considered three benchmark problems: word count, intervened index, and sorting an array. We first compared the performance of our CellCloud with the traditional cloud system. We also considered different failure rate of bidder and measured the performance of our CellCloud system.

We built our CCS on a Desktop with 2.8 GHz Core^{TM} Quad CPU along with 7.7 GB RAM. The Android Standard Development Kit (SDK) on Java eclipse platform was used for implementation and performance analysis of CellCloud Central System. Android SDK is proprietary software from Google, which is installed on the open source platform Eclipse for performing computation on android system. In our experimental set up, we considered that CellCloud can have a maximum of 100 base stations, where a maximum of 1000 bidders can participate under each base station. Each bidder can have any of the three types of mobile devices. The rating point of each bidder ranges between 0.1 to 1. Each bidder is assigned a failure value ranging from 0.1 to 1. Moreover, the bidders base task completion time is also initialized by generating a random number within a pre-defined range that we derived from the sample run on actual devices.

For network setup, we assumed that the CellCloud is implemented on LTE network structure [30]. We used the NS-3 LTE module for simulating the distribution and transfer of the task to the bidders and receiving the result from them. We used the standard LTE format of 1460 bytes packet size. In LTE, each of the 3 sectors of a base station can transfer data at a rate of 3.3 Gbit/s [31]. We considered a 60-meter cell size, where the UE nodes (bidders) are distributed and served by an end node (Base Station). We considered 1500 bytes maximum transfer unit (MTU) along with propagation delay or channel delay of 0.010 seconds. Interval between two consecutive packets was considered as 0.006 seconds. For simplicity, we assumed that the bidders are not moving.

We also set up a private backend cloud system by creating micro-instances from Amazon EC2 cloud on pay as you go basis, which costs 1.3 cents/hour for storage and 1 cent/GB for data transfer. For our convenience, we referred both the rented bidders and computers as nodes.

11.2 EXPERIMENTAL RESULTS.

Task Completion Time and Cost. First, we selected a word count task of size 1 MB. Initially, we considered that the private cloud is made of only a single node and calculated the task completion time. In amazon pay as you go service, a PC has to be rented for at least an hour. Amazon charges 1.3 cents/hour for a micro instance. However, in our experiment, we considered two scenarios. In the first scenario, we calculated the cost based on the actual time a PC is rented. Therefore, if a PC is rented for 10 minutes, the cost will be 0.13 cents instead of 1.3 cents. In the second scenario, no matter how long a PC is rented, the cost will be calculated on hourly basis. Thus, the cost of hiring a PC for 90 minutes will be 2.6 cents. In each iteration, we added a PC until the task completion time reached the lowest value.

Next, we ran the same task in our CellCloud and measured the task completion time. We used the power meter to determine the power drop on a node in every ms while computing a task. We found that the power drop is approximately 1000 mw. For each node, we calculated the total power drop based on the time that the node is used for computation. From the experimental result of Huang et al. [32], we knew that in LTE technology, the power drop for receiving data at a rate of 1 Mbps is 1340.01 mw in every ms while sending data at the same rate has power drop of 1726.43 mw/ms. We calculated the power drop in every ms for this varying data rate. Based on the time a node is used for sending and receiving data, the total power drop in every node was measured. On an average, the cost of electricity in USA is 8.75-cent/KW hour. Considering the above facts, we measured the cost of hiring bidders for the whole task. From the survey of Brain et al., we found that in best case, the operators can make 200% profit on their total investment if they provide internet services with just 1.9 cents per gigabyte [33]. However, in the worst case, they need to provide Internet services with 8.3 cents per gigabyte to make 200% profits. Considering the amount of data sent and received during the mapping and reducing process, we measured the cost in both the best and worst condition. The resulting cost was computed by adding up this network cost with the bidder hiring cost. We repeated the whole process for 10 times and took the average. We repeated the whole process by increasing the file size by 1 MB in each iteration until the file size is reached 20 MB. Applying the similar procedure, we computed the task completion time for inverted index and sorting problem on both CellCloud and traditional cloud. The time for different task size for each of the three benchmarks is depicted on Figures 4, 5, and 6 respectively. The corresponding cost for task completion is shown in Figures 7, 8, and 9 respectively. We used a logarithmic Y-axis of base 2 for our convenience.

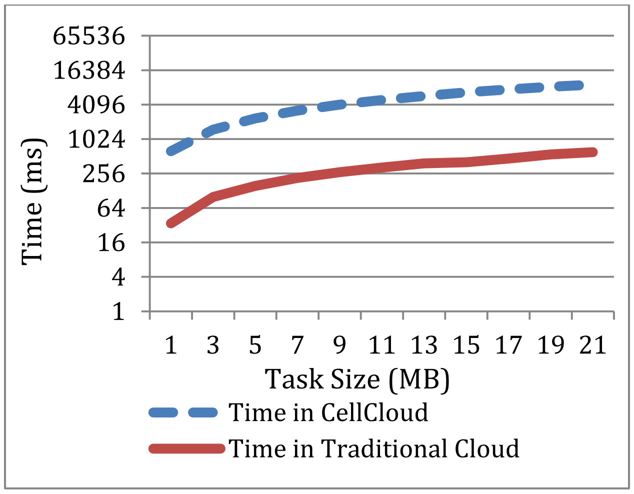

Figure 4. Relationship of Task Completion Time With Increased Task Size for CellCloud and Traditional Cloud for Word Count Problem

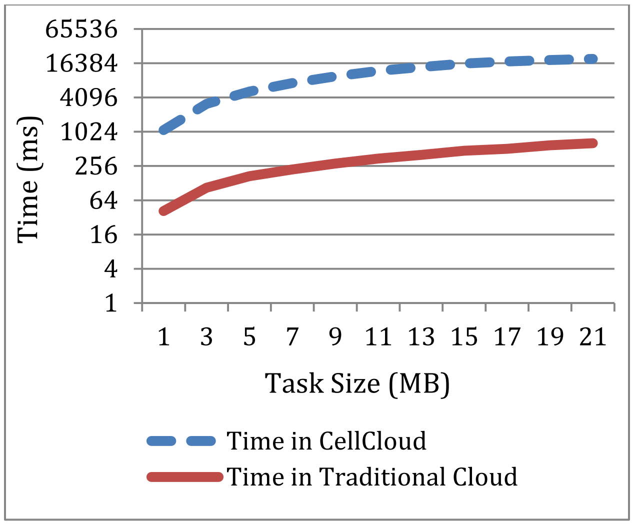

Figure 5. Relationship of Task Completion Time With Increased Task Size for CellCloud and Traditional Cloud for Inverted Index Problem

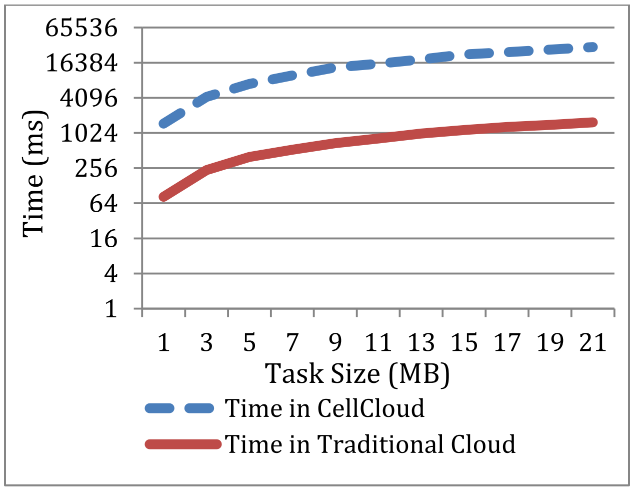

Figure 6. Relationship of Task Completion Time With Increased Task Size for CellCloud and Traditional Cloud for Sorting Problem

From Figures 4, 5, and 6, we notice that the on an average, the task completion time in CellCloud for word count problem is 15 times more than traditional Cloud. For inverted index problem, CellCloud takes much higher around 30 times more than traditional cloud. The sorting problem takes approximately 20 times more in CellCloud than traditional cloud. Moreover, according to the first scenario of traditional cloud, Figure 7, 8, and 9 show that the cost is approximately half compared to even the best case scenario of CellCloud for word count and inverted index problem. For sorting problem, traditional cloud is 10 times less expensive than the best case scenario of CellCloud.

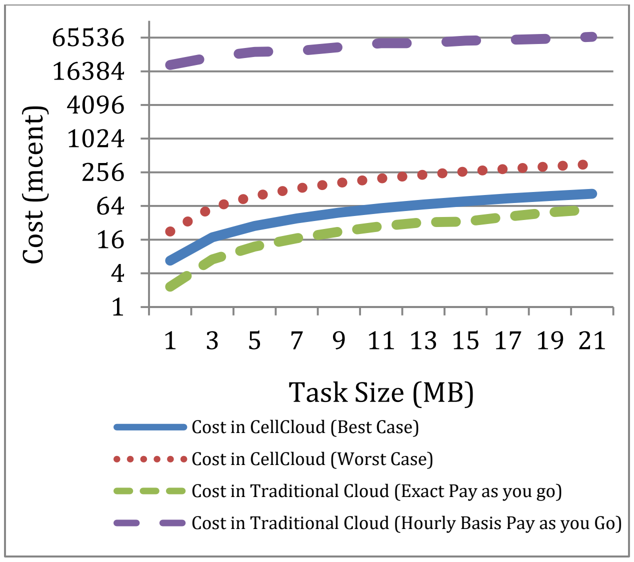

Figure 7. Relationship of Cost with Increased Task Size for CellCloud and Traditional Cloud for Word Count Problem

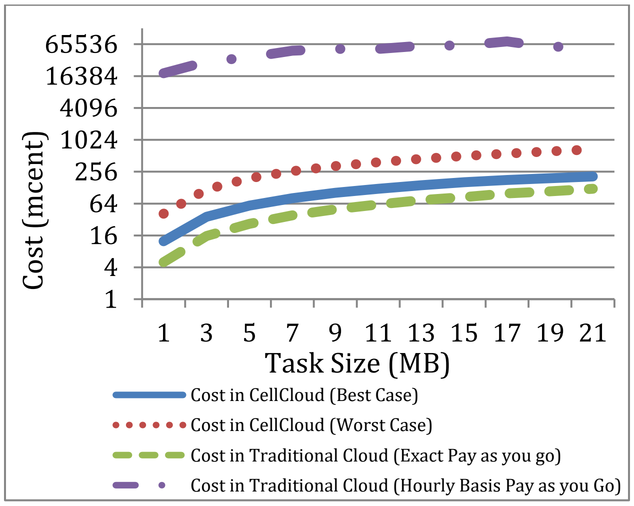

Figure 8. Relationship of Cost with Increased Task Size for CellCloud and Traditional Cloud for Inverted Index Problem

On the other hand, Figure 7 and 8, and 9 depict that the cost for word count, inverted index, and sorting problem, according to the second scenario of traditional cloud, is approximately 270, 150, and 90 times more than the cost of CellCloud in the worst case. However, if we observe the pricing policy of the major cloud providers, then the second scenario is more practical. Therefore, our CellCloud system might not ensure lowest time but it is far less expensive than traditional cloud. Let us consider the ultimate gain as the ratio of gain in cost to gain in time. The ultimate gain with CellCloud is 5 times more compared to traditional cloud for both inverted index and sorting problem and that of word count problem is approximately 15 times.

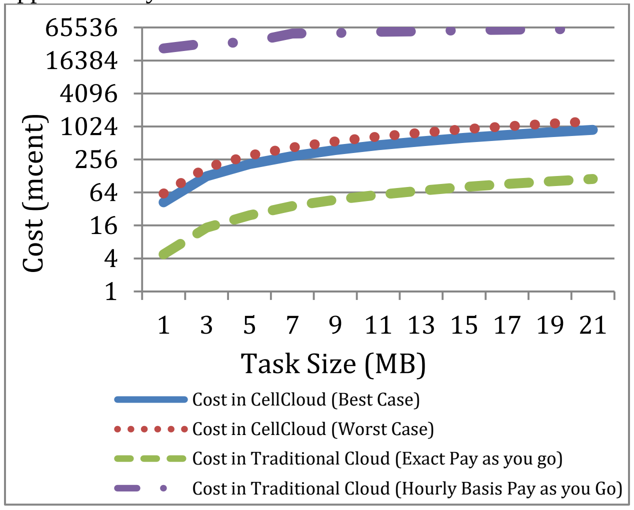

Figure 9. Relationship of Cost with Increased Task Size for CellCloud and Traditional Cloud for Sorting Problem

Computations with Node Failures. In our next experiment, we assumed that the bidder can fail anytime during task execution. The probability of failure for each bidder can varies from 0% to 100%. During our simulation, we assigned a failure value for each node under a base station. We also fixed a threshold for failure and assumed that the nodes whose assigned failure value is below the threshold will fail to perform the task on time. Each base station has the information of possible task completion time for each bidder. If the base station does not receive result within that time it will find another bidder and send the task to that bidder. The bidder that fails will not receive money for the task. The variation of task completion time for different failure rate is depicted in Figures 10, 11, and 12.

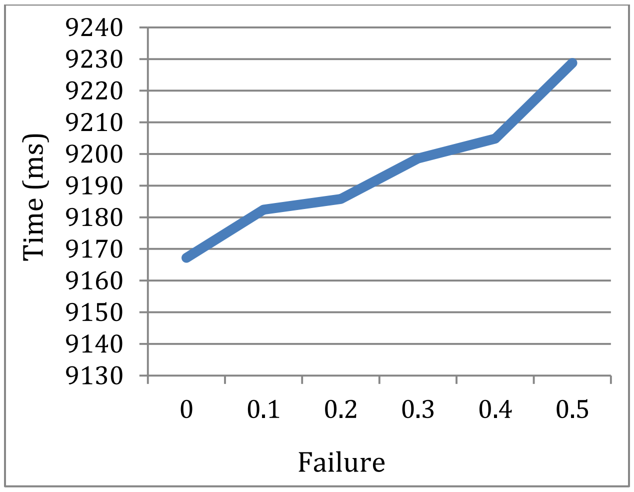

Figure 10. Relationship of Time with Increased Failure for 20 MB Word Count Task

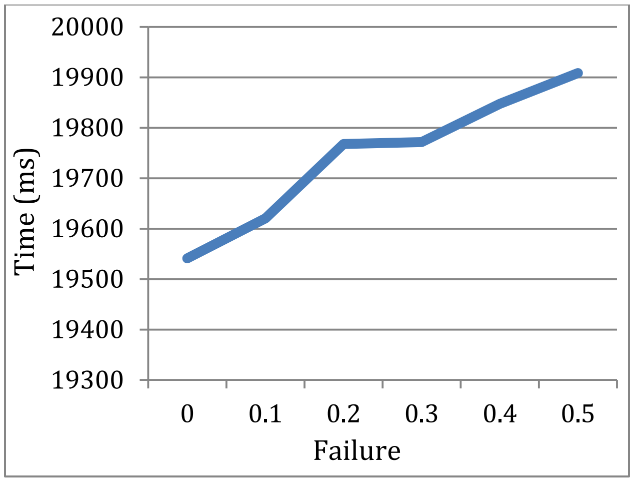

Figure 12. Relationship of Time with Increased Failure for 20 MB Inverted Index Task

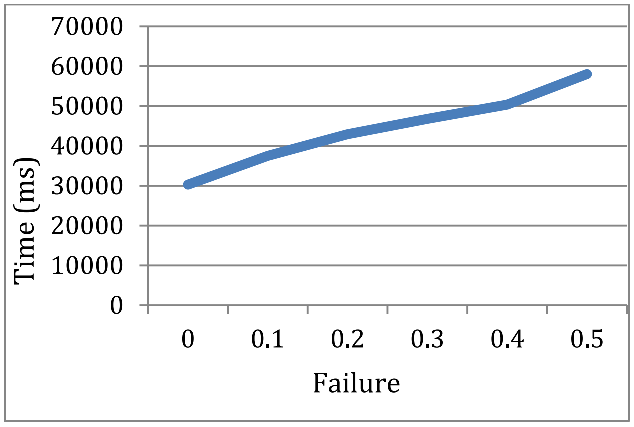

Figure 12. Relationship of Time with Increased Failure for 20 MB Sorting Task

From Figures 10, 11, and 12, we notice a very little increase in task completion time with increasing failure rate. For word count and inverted index problem, the rate of increment in task completion time is 1 and 2 percent respectively from failure value 0 to failure value 0.5. The increasing rate is small because we are distributing the task among a large number of bidders. Hence, each bidder receives a very small portion of the task. Therefore, failure of a bidder does not affect much on overall task completion time. However, for sorting problem, we identified approximately 200% increase in time from failure rate 0 to 0.5. Each node in sorting problem takes much larger time even for a very small portion of the task. Moreover, the input and output size is the same in sorting problem. Thus, the base station has to wait for much longer time before transmitting the task again to a different node. However, considering failure rate of 0.5, the increase in time is still much smaller as compared to the overall task completion time.

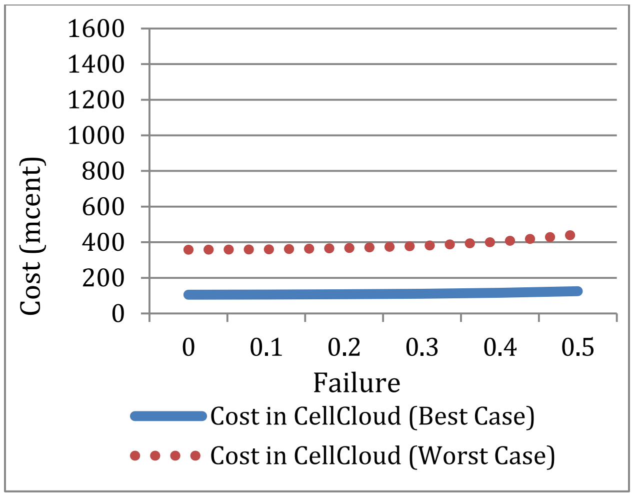

Figure 13. Relationship of Cost with Increased Failure for 20 MB Word Count Task

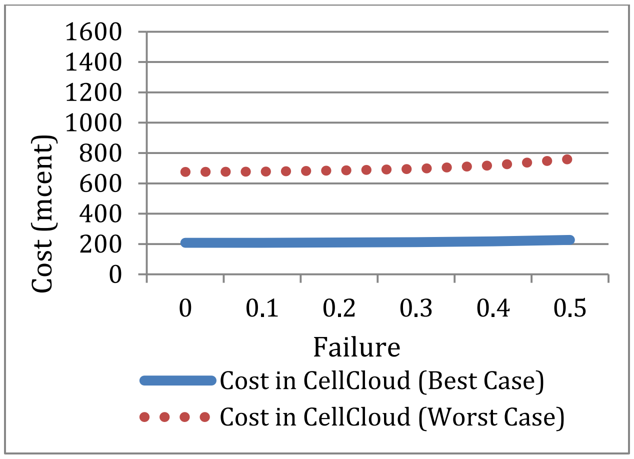

Figure 14. Relationship of Cost with Increased Failure for 20 MB Inverted Index Task

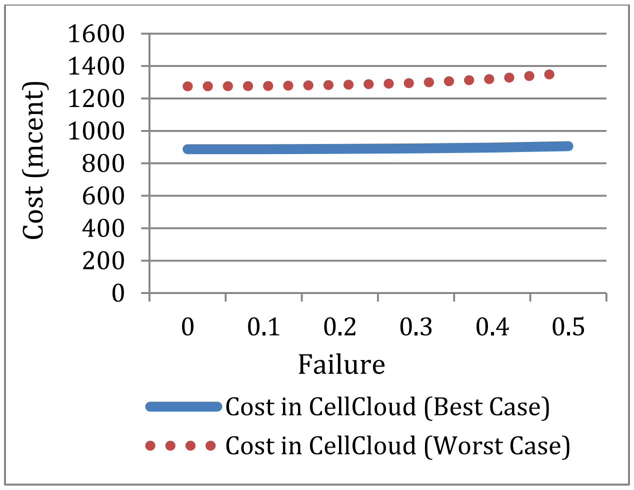

Figure 15. Relationship of Cost with Increased Failure for 20 MB Sorting Task

On the other hand, if we consider Figures 13, 14, and 15, we notice that the cost increases very slightly with increasing rate of failure. From failure rate 0 to 0.5, in best case, the cost increases approximately 18%, 10%, and 2% for word count, inverted index, and sorting problem respectively. The increase rate is approximately 24%, 12%, and 7% for word count, inverted index, and sorting problem respectively. In our CellCloud architecture, the bidders will not receive any money if they fail to perform the task on time. Hence, when a bidder fails we just need to consider the network cost for sending the task to the bidder.

12. FUTURE OF CELLCLOUD

12.1 Speech Recognition.

In several scenarios we might need to recognize the speech [34]. For example if we move to a place where people speak different languages then we might feel difficulty to communicate with them. In such case we might want our mobile device to act as a translator. In addition, sometimes we meet a person after long time and forget the name of that person. Using speech recognizer, we can match the voice of the person with our previously recorded voice and identify that person. However, it requires a substantial computational power and a battery power for running the tool. Instead, our mobile device can record the voice and transmit the speech signal to the cloud. The cloud returns the recognition result after processing.

12.2 Image Analysis.

Image analysis might require when we want to know the details of a visible image. For example, often we visit a museum where we see different types of arts. We are always enthusiasm to gather information about those arts. We might not want to run a complex image analysis tool on our limited power mobile devices. The picture of those arts can be sent to the cloud for analyzing. The cloud sends the detail information of the art back to the user after the processing.

12.3 Real Time Information Gathering.

People might be interested to gather information quickly about a specific area with minimal amount of cost. For example, we might want to go to a party and want to know the suitable dress for the party. Moreover, we might be also interested to know the number of people so far in that party. We can easily get that information using CellCloud. The CCS can ask the bidders on that party to collect and process the required information.

12.4 Face Recognition.

Face recognition is a widely used for applications such as surveillance, airport security, law enforcement, and border patrol [35]. First, an input image is taken for analysis by the face recognition algorithm. From the result of the analysis, we extract several information such as, size, shape, and position etc. of various facial features. These extracted features are compared with the images existed on an already existing facial database for a match. However, this complex face recognition algorithm is hard to run on a user limited mobile resource. The user can acquire the image and send it to the cloud for processing.

13. Discussion

From our simulation result, we see that the task completion time in CellCloud is much higher than traditional cloud. This might lead to a question why will we use CellCloud? If clients want immediate result of a problem, then CellCloud will not be the right choice for them. However, sometimes clients want service where waiting time does not matter much. In fact, clients are more concern about the cost and complexity. In such cases, CellCloud could be a better option for them than traditional cloud. They are already connected to a mobile network and therefore, requesting for cloud service from the same mobile network provider is much easier than login to traditional cloud provider and ask for service. Moreover, they are getting the service with much cheaper rate than using the traditional cloud.

If we further analyze the task completion time of our CellCloud then we can see that among the total task completion time 80% time is required for sending the task to the bidders and receive the result back to CCS. Even though the data transfer rate in 4G networks is very high, it is still much smaller than the data transfer rate inside the traditional cloud such as Amazon, Google etc. The task completion time in CellCloud and Traditional Cloud will be approximately same if we do not consider the task transfer time or if we consider the similar data rate in both cases. With the advancement of technology, the data transfer rate in mobile network is increasing every couple of years. Researchers have already started working of the improvement of mobile network that will soon converge to the 5G technology [36]. According to the specification defined in the blueprint of 5G technology, the data transfer rate will be much higher around 10Gb/s as compared to 3Gb/s 4G technology. Moreover, 5G technology will have very small amount of latency and response time, and will consume much smaller power than the currently using 4G technology. If we deploy our CellCloud according to the specification defined in 5G technology, the task completion time will be reduced to approximately 1/3 of the current task completion time. In that case, CellCloud can also serve the task, which requires immediate result with significantly smaller amount of cost than traditional cloud.

14. Conclusion and Future Work

The current trend of rapidly growing number of smart phone users along with the tendency of switching to new phones in every couple of years is creating a big pile of unused mobile devices. To the best of our knowledge, CellCloud is the first protocol that attempts to reshape the definition of mobile cloud by incorporating these unused but available resources. Along with the detailed architecture of such a system, we have developed a cost model to analyze the benefits from both mobile owners' and provider's point of view. CellCloud features, such as, facilitating different pricing options for different deadlines and level of reliability, providing money to the mobile owners for sharing their unused resources, and lessening operational cost compared to traditional cloud for the cloud provider ensure that such a model can create a win-win situation for all the parties. Currently, we are trying to build a model by which, the CellCloud provider can provide an estimation of task completion time and the associated cost before accepting a task from the client. For doing this, we are planning to train our system with sample tasks of various sizes. Based on the result obtained from the training, we will develop a map between task size and the required amount of resource. We also plan to deploy the architecture in small scale in real life mobile infrastructure to analyze the feasibility of the model.

15. References

[1] Cynthia, "The ongoing investment and hidden costs? make cloud solutions more expensive over time," http: //www.contentmanagement.com/cloud-computing-myth-2, September 2011.

[2] Cade Metz, "Why some startups say the cloud is a waste of money," http://www.wired.com/2013/08/memsql-and-amazon, August 2013.

[3] D. Kawamoto, "Amazon, microsoft cloud services could suffer from shutdown," http://news.dice.com/2013/10/04/ amazon-microsoft-cloud-services-could-suffer-from-shutdown-099, October 2013.

[4] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," Wireless Communications and Mobile Computing, pp. n/a–n/a, 2011.

[5] M. R. Prasad, J. Gyani, and P. R. K. Murti, "Mobile cloud computing: Implications and challenges," ISSN 2225-0506, vol. 2, no. 7, pp. 7–15, 2012.

[6] James, "Lookout finds there are over 28 million old mobile phones going unused in the uk," http://www.tracyandmatt.co. uk/blogs/index.php/lookout- finds- there- are- over- 28- million, November 2012.

[7] M. Alzain, B. Soh, and E. Pardede, "A new approach using redundancy technique to improve security in cloud computing," in International Conference on Cyber Security, Cyber Warfare and Digital Forensic, 2012, pp. 230–235.

[8] A. Bendahmane, M. Essaaidi, A. El Moussaoui, and A. Younes, "Result verification mechanism for mapreduce computation in- tegrity in cloud computing," in International Conference on Complex Systems, 2012, pp. 1–6.

[9] H. A. Lagar-Cavilla, J. A. Whitney, A. M. Scannell, P. Patchin, S. M. Rumble, E. de Lara, M. Brudno, and M. Satyanarayanan, "Snowflock: rapid virtual machine cloning for cloud computing," in Proceedings of the 4th ACM European conference on Computer systems. ACM, 2009, pp. 1–12.

[10] J. H. Christensen, "Using restful web-services and cloud computing to create next generation mobile applications," in Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications. ACM, 2009, pp. 627–634.

[11] B. Chun and P. Maniatis, "Augmented smartphone applications through clone cloud execution," in Proceedings of the 12th conference on Hot topics in operating systems. USENIX Association, 2009, pp. 8–8.

[12] I. Giurgiu, O. Riva, D. Juric, I. Krivulev, and G. Alonso, "Calling the cloud: enabling mobile phones as interfaces to cloud applications," in Proceedings of the ACM/IFIP/USENIX 10th international conference on Middleware, 2009, pp. 83–102.

[13] X. Luo, "From augmented reality to augmented computing: A look at cloud-mobile convergence," International Symposium on Ubiquitous Virtual Reality, vol. 0, pp. 29–32, 2009.

[14] E.E.Marinelli,"Hyrax:cloudcomputingonmobiledevicesusing mapreduce," DTIC Document, Tech. Rep., 2009.

[15] E. Miluzzo, R. Ca´ceres, and Y.-F. Chen, "Vision: mclouds - computing on clouds of mobile devices," in Proceedings of the third ACM workshop on Mobile cloud computing and services, 2012, pp. 9–14.

[16] S. Noor, M. Haque, and R. Hasan, "Cellcloud: A novel cost effective formation of mobile cloud based on bidding incentives," in Proceedings of the 7th IEEE International Conference on Cloud Computing (IEEE Cloud), 2014.

[17] A. Shye, B. Scholbrock, G. Memik, and P. A. Dinda, "Characterizing and modeling user activity on smartphones: summary," in Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems, 2010, pp. 375–376.

[18] A. Mahesri and V. Vardhan, "Power consumption breakdown on a modern laptop," in Power-Aware Computer Systems. Springer Berlin Heidelberg, 2005, vol. 3471, pp. 165–180.

[19] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," in Proceedings of the USENIX annual technical conference, 2010, pp. 21–21.

[20] A. Beberg, D. Ensign, G. Jayachandran, S. Khaliq, and V. Pande, "Folding@home: Lessons from eight years of volunteer dis- tributed computing," in IEEE International Symposium on Parallel Distributed Processing, 2009, pp. 1–8.

[21] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: the condor experience," Concurrency and Computation: Practice and Experience, vol. 17, pp. 323–356, 2005.

[22] E.Korpela,D.Werthimer,D.Anderson,J.Cobb,andM.Leboisky, "Seti@home-massively distributed computing for seti," Computing in Science Engineering, vol. 3, pp. 78–83, 2001.

[23] S. Dashti, J. Reilly, J. D. Bray, A. Bayen, S. Glaser, M. R. Ervasti, and N. Davies, "Using personal devices to deliver rapid semi- qualitative earthquake shaking information," GeoEngineering Re- port, 2011.

[24] A. Mednis, G. Strazdins, R. Zviedris, G. Kanonirs, and L. Selavo, "Real time pothole detection using android smartphones with ac- celerometers," in International Conference on Distributed Computing in Sensor Systems (DCOSS), June 2011, pp. 1–6.

[25] P. Angin, Bharat, and K. Bhargava, "Real-time mobile-cloud computing for context-aware blind navigation."

[26] D. Hoang and L. Chen, "Mobile cloud for assistive healthcare (mocash)," in IEEE Asia-Pacific Services Computing Conference (AP-SCC), Dec 2010, pp. 325–332.

[27] H. Falaki, R. Mahajan, and D. Estrin, "Systemsens: a tool for monitoring usage in smartphone research deployments," in Proceedings of the sixth international workshop on MobiArch, 2011, pp. 25–30. [27] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," in Proceedings of the 2009 conference on Hot topics in cloud computing,, 2009.

[28] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," in Proceedings of the 2009 conference on Hot topics in cloud computing,, 2009.

[29] D. G. Murray, G. Milos, and S. Hand, "Improving xen security through disaggregation," in Proceedings of the fourth ACM SIG-

PLAN/SIGOPS international conference on Virtual execution environments, 2008, pp. 151–160.
[30] I. F. Akyildiz, D. M. Gutierrez-Estevez, and E. C. Reyes, "The evolution to 4g cellular systems: Lte- advanced," Physical Communication, vol. 3, no. 4, pp. 217 – 244, 2010.
[31] C. Dalela, "Article: Prediction methods for long term evolution (lte) advanced network at 2.4 ghz, 2.6 ghz and 3.5 ghz," IJCA Proceedings on National Conference on Communication Technologies & its impact on Next Generation Computing 2012, vol. CTNGC, no. 2, pp. 9–13.
[32] J.Huang,F.Qian,A.Gerber,Z.M.Mao,S.Sen,andO.Spatscheck, "A close examination of performance and power characteristics of 4g lte networks," in Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services, ser. MobiSys '12. New York, NY, USA: ACM, 2012, pp. 225–238.
[33] M. Brain, "What does a gigabyte of internet service really cost? a look at the worst case scenario," http://goo.gl/FsINu, April 2011.
[34] Y.-S. Chang, S.-H. Hung, N. Wang, and B.-S. Lin, "Csr: A cloud-assisted speech recognition service for personal mobile device," in 2011 International Conference on Parallel Processing (ICPP), Sept 2011, pp. 305–314.
[35] T.Soyata, R.Muraleedharan, C.Funai, M.Kwon, and W.Heinzelman, "Cloud-vision: Real-time face recognition using a mobile- cloudlet-cloud acceleration architecture," in 2012 IEEE Symposium on Computers and Communications (ISCC), July 2012, pp. 000 059– 000 066.
[36] "5G: A technology vision," http://www.huawei.com/ 5Gwhitepaper, 2013.

Authors

Shahid Al Noor is a PhD student and a graduate teaching assistant at Computer and Information Science at UAB. His research interest is in Mobile Cloud Computing, Mobile Device Security, and Cloud Security. He received his B.Sc. and M.Sc. in Information and Communication Engineering from University of Rajshahi, Bangladesh in 2006 and 2007 respectively.

Dr. Ragib Hasan is a tenure-track Assistant Professor at the Department of Computer and Information Sciences at the University of Alabama at Birmingham. Prior to joining UAB, He received his Ph.D. and M.S. in Computer Science from the University of Illinois at Urbana Champaign in October, 2009, and December, 2005, respectively, and was an NSF/CRA Computing Innovation Fellow post-doc at the Department of Computer Science, Johns Hopkins University. Hasan has received multiple awards in his career, including the 2014 NSF CAREER Award, 2013 Google RISE Award, and 2009 NSF Computing Innovation Fellowship. His research interest lies in the area of efficient and secure systems including secure clouds, mobile devices, and financial computing systems.

Md Munirul Haque is currently working as Research Scientist in Regenstrief Center for Healthcare Engineering, Purdue University. He received his Ph.D. from Marquette University, USA in 2013 before joining SECuRE and Trustworthy Computing Lab (SECRETLab) at the University of Alabama at Birmingham as post-doctoral fellow. His academic studies and training have made him an expert in the area of mobile based real life applications. He is the recipient of the Ross Fellowship Award for outstanding Ph.D. student and several best paper (COMPSAC 2007, CHI 2012, RACS 2013) and poster awards. His field of interest encompasses security and privacy in pervasive health, m-Health, mobile cloud, and HCI (Human Computer Interaction).