MobiShare : Uber for Mobile Computing

Anurag Ekkati
Ashwin Ramakrishnan
Hanumath Sai Srikanth Vidyasagar Sadhu

Abstract

We are heading towards a world where Internet connectivity is ubiquitous with IPv6 threatening to infiltrate into every square inch of the present explored Earth. Even with this rise of Internet adoption we think that there will be many places where mobile ad-hoc networks with no internet connectivity will play a dominant role. We visualize a scenario where a user may want to exploit the processing capability of other devices in order to supplement his processing power for different uses like data analytics, data processing (e.g., image recognition) or distributed data storage.

Our model is analogous to Uber business model, where we are trying to utilize the existing and unused computational resources to their maximum. Currently, the number of mobile devices are more than the world population and by the end of 2017 it is expected to become 1.4 devices per user [1]. But, many of these devices are working below their optimum capabilities? We know that many of them are lying idle. Harnessing the unused mobile computation power of these mobile devices by forming mobile-ad-hoc computing networks to collectively perform some useful task is our main goal. To realize this goal, we propose a model (mobile application) that advocates network creation, task distribution and incentive payments.

I. Introduction

I. Motivation

The amount of computational power that is getting embedded into the mobile devices is increasing at a faster rate every year. Mobile devices are currently deployed with dual and quad core processors with clocks speeds in the range of 1-2 Ghz and RAM capacities of 2-3 GB. In addition to this, their storage capacity is also continuously increasing with the current capacity being around 128 GB. But, how many of these devices are utilizing these resources to their maximum. Our project aims at harnessing the unused resources of these mobile devices, by providing incentives to the devices which are contributing their resources.

The devices which agree to share their resources, for executing a particular job, forms a mobile ad-hoc network. We can perceive this as a mobile cloud with computing capabilities. The main motivation is to create such computing clouds which can do the requisite jobs at a much cheaper and efficient way when compared to existing distributed cloud infrastructure (Yes, we mean Amazon et al).

II. Related Work

A mobile cloud consists of a network of mobile devices which are capable of collectively executing a task. It can be used to realize real-time, in-situ processing based application such as image recognition . [2] developed a framework making use of mobile clouds to run such applications. The framework consists of three main entities – arbitrator, service requester and service providers, all of them being the mobile devices. A service requester which has certain task to be performed contacts the arbitrator who then distributes the work among the service providers and returns the result back to service requester. One of the drawbacks of this framework is the lack of incentives for service providers to execute the task. Also, this framework doesn't consider connection reliability of a node. It assumes all the nodes are always available to perform the task. Another drawback is the centralized arbitrator model which is a single point of failure. We address all these challenges in our new framework. [3] realizes the traditional cloud computing on mobile devices by distributing the task to the devices through base stations. This also doesn't consider the uncertainty arising from disconnections of the mobile devices and is not amenable for real-time in-situ processing based applications.

III. Proposed Scheme

Our model which will be available as an application (we call it *MobiShare*) can be downloaded for free from appstore. It consists of four major functionalities discussed briefly below and elaborated in detail in further sections:

1. Scoring Algorithm: In our model mobile devices are assigned a score which shows the device's reliability in performing a given task. The score is calculated based on these factors

a) Durations of past connections
b) Battery power
c) Processing power
d) Storage capacity

This concept of score is fundamental to our model and is used (in various forms) in many scenarios explained below.

2. Arbitrator selection: Once MobiShare is installed and opened on a mobile device, it detects if a network is already formed by checking for packets which have the MobiShare options checked in the packet options and if there is no such network, it creates a network. Once the network is formed, an arbitrator is elected based on a predefined algorithm. Arbitrator maintains the state of all nodes in the network including their score and activity status.

3. Incentives: Devices earn points for the amount of work they do and they can later redeem them for money after reaching certain threshold (They can't encash their rewards until they reach a threshold number of points).

4. Task Distribution: Whenever a device has a task to be executed, it contacts the arbitrator to check all the devices that are available to perform tasks. It then decides which devices to choose based on their score (and hence, indirectly their pay-rate) keeping in mind the urgency with which it needs the job to be performed.
When the app is opened, the events 2, 1, 3, 4 are executed in that order. Also, as of now, we assume that the app is secure and its data cannot be tampered by the mobile owner or some other third party.

II. Proposed Scheme

A. Scoring Algorithm

When a device downloads the application, it is given a unique ID which identifies it across multiple connections on the network. The device is categorized based on what kind of service it is willing to extend (viz; storage, processing power)

Since it is a mobile network, it is assumed that the device can join or leave the network according to its will which introduces unreliability to the network. In order bring in a reliability metric (in terms of staying connected to the network), we introduce the concept of giving a 'Score' to every device. For this purpose, we introduce the "duration" metric to identify how long a device was present in the network. This "duration" metric plays a key role (apart from other factors such as battery power, processing power, etc) in assigning the reward score for a device. Based on the reward score, devices are chosen to execute the task as desired by the service requester.

The application in each device maintains the following table for calculating and storing the reward score:

Device ID	**S_0 (mins)**	**S_1 (mins)**	**S_2 (mins)**	**Reward Score**
XYZ	20	10	15	100

Where S = Time (in) - Time (out) and the subscript indicates history of past connections with S_0 being the recent.
[The duration for which the device stayed connected to the network]

The reward score is calculated using multiple factors:

a) Duration (S0, S1, S2) : The longer a device stays in the network, better is its reward score. This is a measure of the reliability of the device.
b) Processing Power (PP) : The application has a field to be input by the user indicating how much of the CPU he/she is willing to allocate for this purpose. The more he/she allocates, the more is its reward score.
c) Storage Capacity/Memory (SC) : This also is user selectable in the app. The larger the storage a device offers, the more is its total reward score.
d) Battery Power (BP): Mobile device reward score is proportional to its battery life.
e) Penalty (Py) : Whenever a device disconnects from the network while performing an active work, it is penalized. This penalty score continuously increases by 1 base unit (a pre-chosen value) for every time the device disconnects while leaving the task incomplete and for every 3 reliable task completion we reduce the penalty by 1 base unit.

Reward Score = aS0 + bS1 + cS2 + xBP + yPP + zSC - Py

Here,
S0 = Is the latest duration for which the device was in the network
S1 = Is the duration of the previous session for which the device was in the network
S2 =The duration of the previous to previous session for which the device was in the network.
a,b,c,x,y,z- These are coefficients which give weightage to that particular metric.

A few points to note:

- S0, S1, S2 can be extended to more sessions depending on the accuracy needed.
- Typically a will be High (H), b will be Medium (M) and c will be Low (L). The reason we allocate a high, medium and low coefficient to the session duration is because of two reasons: 1. Latest data is more reliable. 2. We want to give new devices a decent chance of competing for tasks when compared to the older devices since it might have only 1 or 2 session durations entries (S0, S1) thereby reducing unfairness towards them.
- Mobile Devices can be connected to power outlets. This means that it will have a higher ‘x’ coefficient.
- Since battery power and processing power are not uniform across all devices (for eg., Device 1 - 1500 mAh battery, 1 GHz processor; Device 2 - 3000 mAh, 2 GHz, Dual Core), we are thinking of a standard metric that normalizes these differences.
- Privacy, security and reliability are important concerns to be addressed.

The application executes Reward Score algorithm typically at the end of the task to update the above table with new Reward Score.

B. Arbitrator Selection Mechanism

Arbitrator is a device that creates and maintains the network. It has knowledge of the state of all nodes in the network. For this reason, it has to stay longer than any other device in the network which necessitates high battery level and reliability requirements from it.
The device with the highest score which is willing to act as arbitrator becomes the arbitrator. The device with the second highest score acts as the sous arbitrator. It will also maintain the same table as the arbitrator and only when there are changes to the main table does the arbitrator send updates to the sous arbitrator.

The arbitrator & the sous arbitrator maintains the following table which reflects the state of all nodes in the network:

Device ID	Active / Inactive	Reward Score	Service Provider/Requester?	Active / Busy

The arbitrator periodically sends heartbeat messages to the nodes in the network to determine if the device is active or inactive. The last column (Active / Busy) represents if the device is currently engaged in a task or if it is free to be assigned tasks to.

We can consider two scenarios, when a device wants to join a network by opening the application

1. Arbitrator not present: It means it is the only device in the network. So, it will create the network (known to the application) so that the new devices can join. It will currently act as an arbitrator.

2. Arbitrator already present: It means the device is about to join a network which has already been established and there is already an arbitrator present. If this device has a better Arbitrator Selection Score than the existing arbitrator by a considerable margin and is willing to act as arbitrator, it becomes the new arbitrator. All the state information gets transferred to this new device. All the network devices updates the identity of new arbitrator and the previous arbitrator now becomes the default sous arbitrator thereby relinquishing the previous sous chef of its duty.

When many devices open the app simultaneously and there is no arbitrator, the process of selecting arbitrator is as follows:

Each device broadcasts its Score along with a flag indicating if it is willing to act as arbitrator. Since each device receives this information from all the other devices, the application determines the device with the highest Score and also willing to act as arbitrator as the primary

arbitrator. The device with second highest score is selected as the back-up arbitrator. This decision is uniform across all devices and thus the arbitrator is selected.

C. Task Distribution

A mobile device (say M) that has a task to be shared, first contacts the arbitrator to know the state of different nodes in the network (i.e. which devices are currently available and what are their corresponding scores). M will then choose the devices to get its task done depending on the number of points available with it and its other requirements like reliability,urgency). Once the device selection is done, it will communicate this information to the arbitrator which then updates its table. The arbitrator will also notify the selected devices they will be working for M.

We assume that the parent process of the job gets executed in the device whose job need to be done. The subtasks (child processes or the threads) will be shared with the worker devices. For each subtask,the application computes the equivalent work units which will be used to calculate the points it has to distribute. These subtasks are distributed among the devices and are executed similar to *Remote Procedure Call (RPC)/ Remote Method Invocation (RMI)* as in Java.

Once the task is done, M will inform the arbitrator that selected devices free (available) and also pays them their deserved incentives. The application in the respective devices will then calculate the new reward score and also updates it to the arbitrator about this.

D. Incentives

Incentives motivate the device owner to share his/her unused mobile resources for earning points (which can later be encashed). This will encourage the users to join this computing network and volunteer their devices. The application can be freely downloaded and installed and the users can start sharing their resources and earn points. However, if a device has a job that it needs to performed through other devices, it needs to either have sufficient points to distribute the other devices for the work done by them or we give the host device an in-app points purchasing option in-lieu of money. The device then transfers its points to the devices from which it gets its work done.

Number of points to be paid to each of the worker device will be determined by the application disseminating the job. The application divides the task to be done into Quantized Work (QW) units. Each unit is executable in itself thereby eliminating dependency issues.

Number of points to be paid per device = QW * S * RS

Where QW is the amount of quantized work done by the device, S is a scaling factor which controls the proportion of reward score (RS) that is used for points and
RS is the corresponding device's reward score.

The application running in the device will make sure that it can pay all the devices which are working on its job.

Points earned by the devices can be redeemed for cash only after certain threshold is reached, the reason behind this strategy being,

1. It will motivate users to participate more in future to push their score over threshold so that they can encash their points.
2. Allowing redemption of lower monetary points incurs higher transaction charges compared to the redemption amount.

Whenever the device is giving task, it will immediately lose the corresponding points that the work entails. The more important point being that the worker devices only get paid at the completion of the task and after closing the session securely. In both the cases, the application itself calculates the necessary points to be deducted as well as credited to devices.

III. Conclusion

We have come up with a mobile application model that enables mobile devices to share their idle/unused resources for useful task execution and earn money in this bargain. This model is applicable in scenarios where a single mobile device doesn't have the resources necessary to execute a bigger task or it doesn't want to drain its battery power by executing heavy applications .

We have ensured task completion reliability through the use of reward score, which is calculated based on the device performance in the past history. Thereby the master device can select desirable worker devices based on reward score. We also optimized the score calculation so that new devices which do not have previous work history are not penalized.

Last but not least, we believe our application has potential to put scores of unused mobile devices to useful work thereby reducing slack as in Uber Business Model.

IV. Future Work

There are multiple facets that we envision for this project. We can think of many different ways to enhance and supplement the mentioned algorithms and features. Most of what we think are

implementation specific and not an abstract concept that we have so far mentioned.
We enlist a few key items that we think would benefit this concept overall.

1) Implementing MobiShare :
   We have so far managed to elucidate the concept of Incentivized Distributed Ad-hoc Mobile Cloud Computing and in the future we want to work on actually building an app that manages to bring this abstract concept to concrete reality. We will be building an android app to begin with and we plan to venture into iOS app as the next stage.

2) Extending the Mobile Ad-hoc concept :
   We want to make this ad-hoc network connect to the internet and then allow users to request and provide services globally thereby removing the physical barriers that ad-hoc networks pose.
   It might even suffice if only the arbitrator is connected to the internet. We can then forward tasks to the arbitrator who in turn can distribute it to the mobile devices in the present in the network.

3) Strengthening Scores and Algorithms :
   The Reward score algorithm and the arbitrator selection mechanism can be enhanced much more based on advanced mathematics involving statistics that can better estimate the coefficient values in the score and give a balanced structure.

4) Quantizing data to be distributed :
   We mention in the task distribution phase that the task to be outsourced is quantized in discrete functions so that every device can take sub-tasks which are themselves executable. We plan to come up with a definite architecture that will help us in this process of quantization to best utilize the resources of networked devices.

5) Strengthening the Incentive Mechanism :
   The feature that makes our concept feasible and lucrative is the incentive mechanism. We planned to do a detailed study on strengthening the incentive mechanism to entice more people to download our app and put the wheels in motion.

References

[1] Akamai: Live and on demand streaming.
http://goo.gl/ZRnHfF

[2] H. Viswanathan, E.K. Lee, I. Rodero, “Uncertainty-aware Autonomic Resource Provisioning for Mobile Cloud Computing”, in IEEE Transactions on Parallel and Distributed Systems, 2012
[3] Shahid Al Noor, Ragib Hasan, and Md Munirul Haque, “CellCloud: A Novel Cost Effective Formation of Mobile Cloud Based on Bidding Incentives”, IEEE International Conference on Cloud Computing, 2014

Other Reference [not cited in the paper]:

[1] Niroshinie Fernando, Seng W. Loke, Wenny Rahayu, “Mobile Cloud Computing: A survey”, Future Generation Computer Systems, 2013
[2] D. Anderson and G. Fedak, “The Computational and Storage Potential of Volunteer Computing,” in Proc. of IEEE Intl. Symp. on Cluster Computing and the Grid (CCGRID), Singapore, May 2006.
[3] D. Chu and M. Humphrey, “Mobile OGSI.NET: Grid Computing on Mobile Devices,” in Proc. of IEEE/ACM Intl. Workshop on Grid Computing, Pittsburgh, PA, Nov. 2004.
[4] P. J. Darby and N. F. Tzeng, “Peer-to-peer Checkpointing Arrangement for Mobile Grid Computing Systems,” in Proc. of Intl. Conf. on High-Performance Parallel and Distributed Computing (HPDC), Monterey, CA, Jun. 2007.
[5] L. dos S. Lima, A. T. A. Gomes, A. Ziviani, M. Endler, L. F. G. Soares, and B. Schulze, “Peer-to-peer Resource Discovery in Mobile Grids,” in Proc. of the Intl. Workshop on Middleware for Grid Computing (MGC), Grenoble, France, Nov. 2005.
[6] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, “MAUI: Making Smartphones Last Longer with Code Offload,” in Proc. of the Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys), San Francisco, CA, Jun. 2010.
[7] M.-R. Ra, A. Sheth, L. Mummert, P. Pillai, D. Wetherall, and R. Govindan, “Odessa: Enabling Interactive Perception Applications on Mobile Devices,” in Proc. of the Intl. Conf. on Mobile systems, Applications, and Services (MobiSys), Bethesda, MD, Jun. 2011.