Mobile Computing - A Green Computing Resource

He Ba, Wendi Heinzelman
Department of Electrical and Computer Engineering
University of Rochester
Rochester, NY, United States
Email: {ba,wheinzel}@ece.rochester.edu

Charles-Antoine Janssen
HealthyBill
41 rue de Livourne
1050 Brussels
Belgium
Email: ca@cajanssen.com

Jiye Shi
UCB Pharma
216 Bath Road
Slough, SL1 4EN
United Kingdom
Email: Jiye.Shi@ucb.com

***Abstract*—Cloud computing provides an approach to accessing shared computing resources. However, a traditional cloud is composed of powerful but energy-hungry workstations. The growth of the population of mobile devices such as smart phones and tablets provides huge amount of idling computing power. In this paper, we describe the design and implementation of a mobile computing system prototype named GEMCloud that utilizes energy efficient mobile devices (e.g., smartphones and tablets) as computing resources. We evaluate the computing power and energy efficiency of the mobile devices through comprehensive experiments. The results show that a cloud computing system with enough mobile devices working cooperatively is able to save 55% to 98% of the energy consumption of conventional server-based clouds while providing comparable computing speed.**

I. Introduction

Cloud computing provides an approach to accessing shared computing resources as a service. Traditionally, the cloud is a group of powerful computers, e.g., servers, workstations, personal computers, etc. However, the traditional cloud computing system usually focuses on performance rather than energy efficiency. As the use of energy resources has raised global concerns, looking for more energy efficient approaches to providing computing power is an urgent task for researchers.

Nowadays, mobile devices such as smartphones and tablets are becoming increasingly powerful and rising quickly in popularity. According to International Data Corporation (IDC)'s statistics [1], 494 million smartphones were sold worldwide in 2011, surpassing the 353 million total sales of PCs. From 2010 to 2011, the sales of smartphones reached an annual growth of 62%, with expected continued increases in sales in the future. Tablet device sales are also rising sharply, jumping from 19 million in 2010 to 69 million in 2011, an astonishing annual growth of 263%. These comparisons are illustrated in Fig. 1. With the continuous growth of annual sales and the evolution of technology, it is reasonable to expect that in the near future there will be enormous amount of computing power available from tablets and smartphones all over the world.

In addition, unlike personal computers, mobile devices are rarely powered off, even when the owners are sleeping, which translates into hours of unutilized computing resources. There is great potential if we can make use of these idle computing resources. However, the approach to utilizing mobile devices for cloud computing has not been researched extensively, leaving the question as to whether a mobile computing system can be powerful and energy efficient at the same time.

In this paper, we investigate and develop a system, named GEMCloud (Green Energy Mobile Cloud) that uses mobile devices to provide distributed computing services to support computationally-complex and parallelizable applications. Besides the implementation, our focus is on the evaluation of the computing capability and the energy efficiency of the system.

The rest of this paper is organized as follows. In Section II, we review the current state of the art in the area of mobile computing. Section III introduces our mobile computing system followed by performance evaluations in Section IV. Finally, Section V concludes the paper and discusses future directions for this research.

II. State of The Art

We propose an energy efficient cloud computing system that provides computational resources from distributed mobile devices to the users. The idea of applying distributed computing and cloud computing to mobile devices has been developed in recent years. However, limited by the traditionally low processing speed and small storage space, the focus of much of this research has been on reducing the computational burden of mobile devices.

One method of reducing the computational burden of mobile devices is to set up an agent between the mobile devices and cloud computing resources to provide mobile devices access

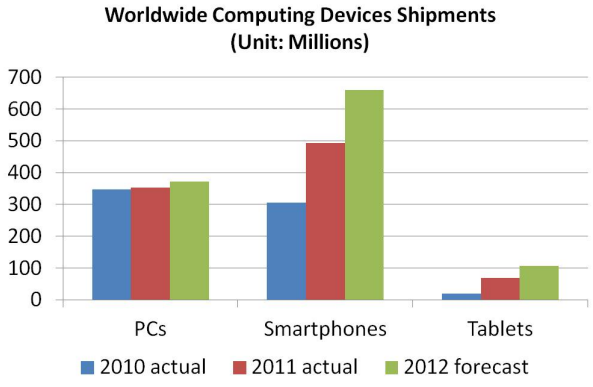

Fig. 1. Computing device sales comparisons. Data are from [1].

978-1-4673-5939-9/13/$31.00 ©2013 IEEE

to the cloud. For example, the Mobile Cloud Middleware (MCM) [2] introduces a middleware framework that manages the connections and communications between mobile phones and clouds. The MCM also provides an asynchronous server-phone communication mechanism that specifically benefits the mobile phone users.

Another approach to reducing the burden of mobile devices is to offload the computationally-heavy executions to the cloud computing resources. For example, CloneCloud [3] is a system that allows a smartphone device to partially offload its application to the phone's clone (an application-level virtual machine) in the cloud. The authors test CloneCloud on an HTC G1 Android phone. Results show that the CloneCloud approach saves execution time and energy consumption on the mobile devices. A similar idea was also investigated by Satyanarayanan et al. [4] and Cuervo et al. [5]. Another example from Chen et al. [6] introduces a framework that allows heavy back-end tasks on an Android phone to be offloaded to an Android virtual machine in the cloud.

Both of the above approaches require the computations to be executed in the cloud, which is composed of dedicated computers. However, the mobile devices themselves could be the source of computing power, too. Recent technological advances have greatly improved the performance of smartphones/tablets in terms of CPU/GPU speed, memory size, and storage space. As an example of one of the fastest tablet on the market, the Asus Nexus 7 [7] is equipped with a $1.2GHz$ quad-core CPU with $1GB$ RAM plus $8GB$/$16GB$ storage. A smartphone or tablet similar to the Asus Nexus 7 can provide a considerable amount of computing power that may be even comparable to the computing power of a desktop computer.

Hyrax [8] has demonstrated the concept of using smartphones as computing resources. The author developed a mobile cloud computing system named Hyrax by porting Hadoop Apache, an open-source implementation of MapReduce, to Android smartphones. Hyrax allows computing jobs to be executed on networked Android smartphones. However, the performance of Hyrax was poor compared with Hadoop on traditional servers, not only because the smartphones were much slower at that time, but also because Hadoop was not originally designed, nor optimized, for mobile devices.

The NativeBOINC for Android project [9] is another example of utilizing mobile devices as computing resources. BOINC (Berkeley Open Infrastructure for Network Computing) [10] is an open-sourced volunteer computing software originally developed for PC users to contribute their computing powers to scientific projects. The NativeBOINC for Android project implemented a BOINC client for Android devices that supports six BOINC projects so far. On the mobile phone client, the user may select several projects to attend and start or stop computing on demand. Eastlack [11] ported BOINC to 4 development boards with various ARM-based mobile processors and compared their performances with Intel processors. The results indicate that mobile processors have energy efficiency advantages over desktop processors. However, this work does not consider system-level performance comparisons. In this paper, we develop a working prototype of GEMCloud, a system that exploits the energy efficiency of mobile devices for processing computationally-complex, parallelizable applications.

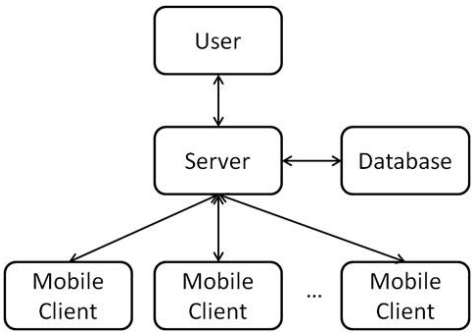

Fig. 2. The mobile computing system architecture.

While the possibility of utilizing mobile devices as computing resources has been demonstrated by researchers, designing and developing a system specific to mobile devices is challenging and must take into account the characteristics of mobile devices in terms of their relatively slow and unstable processing speed, limited battery life, constrained and costly wireless bandwidth and dynamic network topology, among others.

III. The GEMCloud System

In this paper, we design and develop a mobile computing system prototype, namely GEMCloud, that utilizes distributed mobile devices to cooperatively accomplish large parallelizable computational tasks. The main purpose of designing such a system is to find a green approach to making use of the massive amount of idle computing power that is potentially available to the public. In addition, in this paper we show that a mobile computing system has significant advantages in energy efficiency over traditional desktop computing systems, and, therefore, distributed computing on mobile devices should be explored through prototype implementations.

A. System Architecture

Fig. 2 shows the system architecture we utilize to create a distributed mobile computing system. The system consists of a network with users who need computing power, a server that is in charge of the organization of the entire network, a database that records the mobile clients' information and task information, and multiple clients of mobile devices that are the computing resource providers.

The role of the server is to organize the entire network and to coordinate mobile clients to perform computational tasks. The server also maintains a database that stores client information such as each client's unique identification, IP address, hardware capabilities, the tasks each client is performing, project data stored on the device, etc. The server may assign the tasks based on the clients' information. For example, the size and number of tasks assigned to a mobile device may

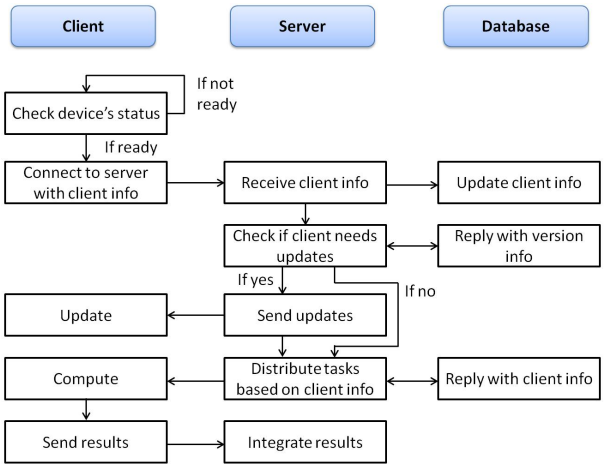

Fig. 3. The server-client protocol flow chart.

be based on the speed and the number of CPU cores on the mobile device.

The clients are mobile devices that connect to the server via the internet and provide computing resources to perform the tasks requested by the user. In our prototype implementation, we use Android [12] smartphones and tablets as the mobile client devices. One of the most important reasons of choosing Android as our development platform is its multi-tasking ability, which is lacking in the iOS platform. Multi-tasking allows an application for mobile devices (i.e., an app) to execute in the background without interfering with the device's other functionalities, which is especially beneficial as multi-core mobile devices are becoming increasingly popular these days. Another important reason to choose the Android platform is that the developer has more control of the mobile device. For example, it is easier to set the percentage of CPU allowed to be used for distributed computing in Android than in iOS.

B. Server-Client Protocols

In our prototype, the clients follow the server-client protocol flow chart described in Fig. 3. Before connecting to the server, the mobile client's application checks the device's status and decides if it should connect to the server and offer its computing resources. For example, if the device is running other applications that consume CPU and memory more than a preset threshold or if the battery is low and not charging, the device will not connect to the server. Such preferences are designed to avoid interfering with the mobile device's normal usage.

After connecting to the server, the client will wait for a response from the server. The server checks the database to determine whether the client's application needs an update. For each mobile client, the server's database stores an entry with a unique client identification, its IP address and other information such as CPU speed, the tasks each client is performing, project data stored on the device, etc. For the prototype we have developed, we are focusing on the evaluation of its computing capability and energy efficiency.

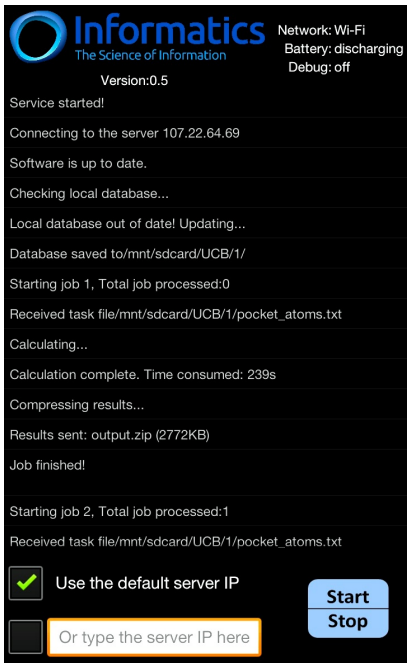

Fig. 4. The prototype screen shot from an Android phone.

Therefore, database management and clients' information that may be used for task distribution are not of concern for now.

Following the updating process, the server assigns tasks to the clients. In our prototype implementation, the server simply assigns the next task on a pre-determined list to the next available mobile device. Again, since our current focus is not on task assignment mechanisms, we do not attempt to optimize task distribution in this paper.

In our prototype, the user's application is a CPU-intensive computing job, which can be split into multiple independent tasks. The user sends a request to the server to complete a job. The server splits the job into small tasks and distributes them to multiple clients. When there are more active clients to provide computing power, the server assigns to each client fewer tasks, which means less time is required for each client to finish the assigned tasks. Therefore, the turnaround time for the job is reduced when more clients contribute to the computation.

Upon reception of the task assignment, the mobile client performs the computation and sends the results back to the server in a compressed file. The server aggregates all the results and sends them back to the user once all the tasks are completed.

IV. Performance Evaluations

The main motivation of developing a mobile computing system in this paper is its potential advantage in energy efficiency. In this section, we evaluate the energy efficiency and computing power of mobile devices and compare them with conventional workstations.

TABLE I
DEVICE SPECIFICATIONS (NOTE: "* 2 (4)" MEANS THERE ARE TWO (FOUR) PHYSICAL CPUS IN THE WORKSTATION).

Device	CPU	Memory	Release Date
Xiaomi Mi-One (MO)	Qualcomm Snapdragon S3 Dual Core 1.5GHz	1GB	2011
Asus Nexus 7 (N7)	Nvidia Tegra 3 Quad Core 1.2GHz	1GB	2012
Samsung Galaxy S3 (GS3)	Qualcomm Snapdragon S4 Dual Core 1.5GHz	2GB	2012
Workstation1 (WS1)	Intel Xeon E7505 * 2 Single Core 3.06GHz * 2	8GB	2003
Workstation2 (WS2)	Intel Xeon X5355 * 2 Quad Core 2.66GHz * 2	24GB	2006
Workstation3 (WS3)	AMD Opteron 6276 * 4 16-Core 2.33GHz * 4	192GB	2011

A. Experimental Setup

In the energy efficiency tests, we evaluate the performance of both the mobile devices and workstations in executing a computationally-complex application. The application was written in C++. The mobile devices we tested all use the Android operating system. Since Android applications are written in Java, we use the JNI (Java Native Interface) to execute the native C++ code on the Android devices. The testing code running on the workstations are directly compiled from the C++ code. This may cause the application for the Android platforms to be slower than that for the workstations due to the overhead of Java and JNI integration. However, this is unavoidable when porting C/C++ applications to mobile platforms and thus represents a realistic situation. We used 3 mobile devices and 3 workstations for testing. Their specifications are listed in Table I.

We measure the power consumed by the mobile devices or workstations using the "Watts up? PRO ES" power meter [13]. According to its specifications, the meter has an accuracy of $\pm 1.5\%$ in terms of wattage measurement. We set the recording interval to the lowest value of $1s$. This provides a good sampling rate for our measurements as the test application takes at least $40s$ even on the fastest workstation. We run the same task 10 times on each device and average the energy consumption readings. For multi-core devices, we run the same task on multiple threads to evaluate the time and energy performance. For mobile devices, we measure the device's energy consumption while the screen is off. For the workstations, we only measure the power consumption of the main unit without the monitor.

B. Experimental Results

The experimental results are shown in Tables II - VII, corresponding to the Xiaomi Mi-One (MO) Android phone, the Samsung Galaxy S3 (GS3) Android phone, the Asus Nexus 7 (N7) Android tablet and the three Linux workstations (WS1, WS2, WS3).

Our first interest on the mobile devices is the computing power they can provide. As expected, the mobile devices are slower than the workstation competitors. The GS3 is equipped with a dual-core CPU and is able to finish 2 tasks in $217.1s$. This means, on average, the GS3 has a computing power of $33.2tasks/hr$. With the quad-core processor, the N7 has the best computing power among all the mobile devices. It is able to complete 4 tasks within $312.4s$, which translates into $46.1tasks/hr$. This performance is close to the WS1's $51.1tasks/hr$, showing that mobile devices now may have computing power comparable to some existing desktops and workstations. The WS1 serves as a convenient benchmark for processing speed. We ran the same task on the Amazon Web Services (AWS) and found that each CPU of the WS1 is roughly equivalent to the AWS m1.small instance, which has 1 EC2 Compute Unit. The very high-end WS3 can accomplish 64 tasks within $50.1s$. In other words, it is able to complete $4598.8tasks/hr$, which is about 98.8 times faster than the N7. Based on the data from Table II - VII, the times required to complete the same number of tasks for all devices can be calculated as shown in Fig. 5.

When the devices are idle, the GS3 has the lowest power

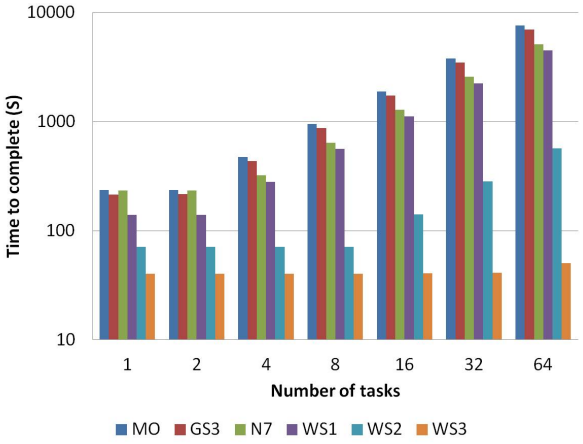

Fig. 5. Comparison of computing time.

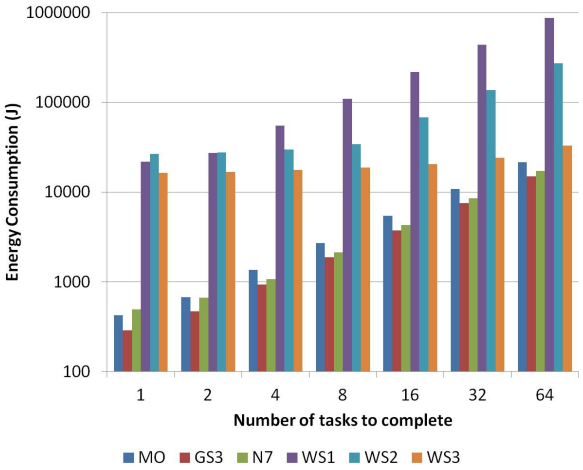

Fig. 6. Comparison of energy consumption.

consumption ($0.001W$) while the MO and N7 consume $0.6W$ and $0.5W$, respectively. As for the workstations we tested, the WS1 has the lowest idle power consumption of $118W$, while the other two workstations consume over $350W$ even when idle. The idle power is an important factor when considering energy efficiency because the power consumed in this period of time is not used for anything productive and is wasted energy. Therefore, in the cases of light utilization, the mobile devices can save a significant amount of energy thanks to their low power consumption when idling.

Due to the fact that mobile device manufacturers build various user interfaces (UIs) on top of the Android platform, the power management strategies vary depending on the device and the manufacturer. For example, the N7 limits the CPU clock while using 4 cores when the screen is turned off. Therefore, we observe a significant increase in the amount of time (25% per core) needed to complete 1 task when all 4 cores are occupied. It is important to consider this power management factor in our future work, as we cannot expect every mobile device to allow applications to run in full speed when the screen is turned off.

As for the energy efficiency, the mobile devices have a clear edge over the workstations. With only 1 task to compute, the GS3 has the best energy efficiency among all the tested devices. It only consumes $288.5J$ to complete a task, while the most energy efficient workstation WS3 requires $16,322J$ to process the same task, 57 times the energy required by the GS3. The WS1 and WS2 consume 75.8 and 92.5 times the energy required by the GS3. In other words, the GS3 saves more than 98% of the energy required by the workstations when a single task is computed.

Both the workstations and the mobile devices have their best energy efficiency when their CPUs are fully loaded. Fig. 6 shows the amount of energy required for each device to complete a certain number of tasks (the energy required for completing more than 64 tasks can be easily calculated based on the measurements in Table II - VII). All the mobile devices we tested have better energy efficiency than the workstations. More specifically, to complete 64 tasks, the GS3 only requires $15,001.6J$, or $234.4J$ per task, which is the least among all the tested devices. The N7 and MO require $17,110.4J$ and $21,648J$, respectively. As for workstations, the WS3 requires $32,981.2J$ in total, which is $515.3J$ per task, 2.2 times what the GS3 requires. The WS1 and WS2 consume $877,721.6J$ and $273,075.2J$ respectively, which is 58.5 and 18.2 times what the GS3 consumes. In other words, the GS3 saves 98.3%, 94.5% and 54.5% of the energy required by the WS1, WS2 and WS3, respectively, when the CPUs of each device are at full load.

In cases where the computing job can be split into multiple independent tasks, the energy efficiency is a performance metric more important than the computing speed, as we can increase the system computing speed by recruiting more devices to work cooperatively on the job. However, there is no easy way to improve the energy efficiency. For example, if a job consists of 320 tasks similar to the ones used in our test, a WS3 takes $250.5s$ to complete the job and consumes $164,906J$ in total. If we use the computing resources of 160 GS3, each processing 2 tasks, the total amount of time required to complete the job is $217.1s$ while the energy consumed is only $74,992J$, 45.5% of that consumed when using WS3.

It should be noted that the WS3 represents a class of high-end workstations/servers with significant hardware acquisition cost, which is rarely seen in commercial cloud computing service offerings. The WS1 and WS2, which uses 58.5 and 18.2 times the energy of what the GS3 costs, respectively, are more representative of servers deployed as part of cloud computing infrastructure. Furthermore, the energy efficiency gap widens quickly as the device utilization level decreases, which is due to the fact that, when being idle, workstations consume hundreds of times more energy than do mobile devices. Therefore, using mobile devices as a green computing resource alternative to a conventional server-based cloud is promising and feasible.

TABLE II
PERFORMANCE RESULTS OF THE XIAOMI MI-ONE (1 CPU, 2 CORES)

	Idle	1 Core 1 Task	2 Cores 2 Tasks
Power (Watt)	0.6	1.8	2.9
Computing Time (Sec)	N/A	235.0	236.1
Total Energy (Joule)	N/A	424.1	676.5
Energy per Task (Joule)	N/A	424.1	338.3

TABLE III
PERFORMANCE RESULTS OF THE SAMSUNG GALAXY S3 (1 CPU, 2 CORES)

	Idle	1 Core 1 Task	2 Cores 2 Tasks
Power (Watt)	0.001	1.3	2.2
Computing Time (Sec)	N/A	214.4	217.1
Total Energy (Joule)	N/A	288.5	468.7
Energy Per Task (Joule)	N/A	288.5	234.4

TABLE IV
PERFORMANCE RESULTS OF THE ASUS NEXUS 7 (1 CPU, 4 CORES)

	Idle	1 Core 1 Task	2 Cores 2 Tasks	4 Cores 4 Tasks
Power (Watt)	0.5	2.1	2.8	3.3
Computing Time (Sec)	N/A	233.7	234.3	320.7
Total Energy (Joule)	N/A	492.3	661.8	1069.4
Energy Per Task (Joule)	N/A	492.3	330.9	267.4

TABLE V
PERFORMANCE RESULTS OF THE WORKSTATION 1 (2 CPUS, 2 CORES)

	Idle	1 Core 1 Task	2 Cores 2 Tasks
Power (Watt)	118.0	156.4	195.8
Computing Time (Sec)	N/A	139.8	140.1
Total Energy (Joule)	N/A	21,864.4	27,428.9
Energy per Task (Joule)	N/A	21,864.4	13,714.4

V. CONCLUSION AND FUTURE WORK

In this paper, we introduce GEMCloud, a mobile cloud computing system that provides computing resources to the

TABLE VI
PERFORMANCE RESULTS OF THE WORKSTATION 2 (2 CPUS, 8 CORES)

	Idle	1 Core 1 Task	2 Cores 2 Tasks
Power (Watt)	362.6	377.9	393.0
Computing Time (Sec)	N/A	70.6	70.6
Total Energy (Joule)	N/A	26,687.8	27,755.1
Energy per Task (Joule)	N/A	26,687.8	13,877.6

	4 Cores 4 Tasks	8 Cores 8 Tasks	
Power (Watt)	423.5	482.2	
Computing Time (Sec)	70.6	70.8	
Total Energy (Joule)	29,917.4	34,134.3	
Energy per Task (Joule)	7479.4	4266.8	

TABLE VII
PERFORMANCE RESULTS OF THE WORKSTATION 3 (4 CPUS, 64 CORES)

	Idle	1 Core 1 Task	2 Cores 2 Tasks
Power (Watt)	395.6	407.8	420.3
Computing Time (Sec)	N/A	40.0	40.0
Total Energy (Joule)	N/A	16,322.0	16,824.4
Energy Per Task (Joule)	N/A	16,322.0	8412.2

	4 Cores 4 Tasks	8 Cores 8 Tasks	16 Cores 16 Tasks
Power (Watt)	438.7	466.9	506.4
Computing Time (Sec)	40.1	40.2	40.6
Total Energy (Joule)	17,586.8	18,751.1	20,538.4
Energy per Task (Joule)	4396.7	2343.9	1283.7

	32 Cores 32 Tasks	64 Cores 64 Tasks	
Power (Watt)	588.8	658.1	
Computing Time (Sec)	41.0	50.1	
Total Energy (Joule)	24,150.8	32,981.2	
Energy Per Task (Joule)	754.7	515.3	

user from energy efficient mobile devices. We provide the design of the system and implement a prototype for testing. Our contribution is mainly focused on the evaluation of the energy efficiency of this system by providing comprehensive tests on the mobile devices. We provide performance comparisons among various mobile devices and workstations. The results show that the smartphones and tablets have lower individual computing power but much higher energy efficiency. The lower computing power can be made up by recruiting more devices, while the energy efficiency is harder to improve given the same type of devices.

The performance measurements demonstrate clearly the potential of using mobile devices as distributed computing resources. Our next step will be to investigate a more sophisticated system design, focusing on task distribution algorithms based on characteristics unique to mobile devices. As mobile devices do not have the same stable environments as workstations have, it will be important to investigate how these random factors affect the overall system performance. The energy and time cost for the communications should be also evaluated and included in the system model. In addition, the security of the system must be investigated to protect the privacy of mobile device owners and to guard against unauthorized access to the computing data and results.

ACKNOWLEDGMENT

This research was funded in part by UCB Pharma and by CEIS, an Empire State Development-designated Center for Advanced Technology. The development of the mobile computing platform prototype was funded by UCB Pharma. The authors thank Dr. Phil Scordis and Dr. Dan Chapman of UCB Pharma for helpful discussions and advice.

REFERENCES

[1] IDC Press Release. [Online]. Available: http://www.idc.com/getdoc.jsp?containerId=prUS23398412

[2] H. Flores, S. N. Srirama, and C. Paniagua, "A generic middleware framework for handling process intensive hybrid cloud services from mobiles," in *Proceedings of the 9th International Conference on Advances in Mobile Computing and Multimedia*, ser. MoMM '11. New York, NY, USA: ACM, 2011, pp. 87–94. [Online]. Available: http://doi.acm.org/10.1145/2095697.2095715

[3] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud," in *Proceedings of the sixth conference on Computer systems*, ser. EuroSys '11. New York, NY, USA: ACM, 2011, pp. 301–314. [Online]. Available: http://doi.acm.org/10.1145/1966445.1966473

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14 –23, oct.-dec. 2009.

[5] 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 *Proceedings of the 8th international conference on Mobile systems, applications, and services*, ser. MobiSys '10. New York, NY, USA: ACM, 2010, pp. 49–62. [Online]. Available: http://doi.acm.org/10.1145/1814433.1814441

[6] E. Chen, S. Ogata, and K. Horikawa, "Offloading android applications to the cloud without customizing android," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, march 2012, pp. 788 –793.

[7] Asus Nexus 7 Android tablet. [Online]. Available: http://www.asus.com/Tablet/Nexus/Nexus_7/

[8] E. E. Marinelli, "Hyrax: Cloud computing on mobile devices using mapreduce," Master's thesis, Carnegie Mellon University, 2009.

[9] Native Boinc for Android. [Online]. Available: http://nativeboinc.org/site/uncat/start

[10] D. P. Anderson, "Boinc: A system for public-resource computing and storage," in *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, ser. GRID '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 4–10. [Online]. Available: http://dx.doi.org/10.1109/GRID.2004.14

[11] J. R. EASTLACK, "Extending volunteer computing to mobile devices," Master's thesis, New Mexico State University, 2011.

[12] Android. [Online]. Available: http://www.android.com/

[13] Watts up? PRO ES Watt meter. [Online]. Available: https://www.wattsupmeters.com/secure/products.php?pn=0