A Survey on Algorithms for Limited Resources Optimization and Utilization by Games on Smartphones

Article *in* International Journal of Computer Applications · June 2016

DOI: 10.5120/ijca2016910262

3 authors, including:

Deborah Natumanya
institute of computer science
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Robert Mugonza
Mbarara University of Science & Technology (MUST)
8 PUBLICATIONS **3** CITATIONS
SEE PROFILE

International Journal of Computer Applications (0975 – 8887)
Volume 144 – No.5, June 2016

A Survey on Algorithms for Limited Resources Optimization and Utilization by Games on Smartphones

Deborah Natumanya
Dept. of Information Technology
Mbarara University of Science and Technology

Nabaasa Evarist
Dept. of Computer Science
Mbarara University of Science and Technology

Mugonza Robert
Dept. of Computer Science
Mbarara University of Science and Technology

ABSTRACT

Smartphones have been characterized by their limited computational resources such as battery life, processor performance and storage capacity. The above mentioned limitations can be overcome by utilizing mobile cloud computing where the smartphone can utilize the sufficient cloud resources. A number of solutions have been suggested and developed to alleviate the issues with smartphone resource limitations however they are not efficient.

This paper provides an overview on the existing algorithms for accessing resource hungry applications on the cloud clearly indicating their techniques and flaws. The paper also describes the directions for the future research.

General Terms

Algorithms for accessing resource hungry games stored on a cloud.

Keywords

Mobile Cloud Computing, Algorithms, Smartphones, Resource Hungry Games

1. INTRODUCTION

Smartphones are the most commonly used mobile devices, they are an advanced version of mobile handsets as they are not only voice centric but also act as mobile personal computers because of the diverse functionalities they offer such as web accessibility, mobile gaming, advances in processor performances and the ability to support virtualization.

Mobile handsets are still resource-limited devices despite the tremendous efforts done by hardware manufacturers and operating system vendors in the last years. Modern mobile platforms such as Android and iPhone are built as modifications of general-purpose operating systems which do not consider energy-efficiency as a key performance goal. In fact, modern handsets incorporate some power-hungry hardware resources such as touchscreen displays and location sensors, and they support Internet data services so they are always connected to the network [1]. All these resources bootstrapped a rich ecosystem of mobile applications but their design is clearly driven by usability factors rather than energy efficiency. Since the mid-90s, researchers have been emphasizing the need of considering energy as fundamental.

Resource hungry applications are software's that require large amounts of resources to run on limited architectures. These resources include storage, processor power and battery power. During the research carried out by AVG App report in 2015 [2], spotify was ranked number 2 resource consuming application followed by the game applications like Farmville, puzzle games, dragons, boom beach and deer hunter. Resource hungry games have been found to be very addictive [3] to the game players. They are characterized by high graphics, large size, require a powerful Graphical Processing Unit (GPU) and Central Processing Unit (CPU), interesting game levels that lure the gamer into continuous playing and all these have pushed the mobile platform to its limits.

Widely recognized as a future computing infrastructure, Cloud computing allows users to access elastic and shared infrastructure (*servers, networks, storage, software, and application services*) for their computing needs at a pay per use basis from cloud providers including:- *Google, Amazon, etc.* As a result, this computing paradigm has enabled access to mobile applications can be ubiquitously provisioned and availed with the negligible management efforts [4]. More so, its scalable support for also services for mobile users and the growing demand for mobile applications, mobile cloud computing has emerged [5], bringing into play new types of computing facilities and application services that enable mobile developers to take full advantage of the cloud. Mobile cloud developers can then rely on a model where application computation (data storage and processing) happen outside the mobile device [6], [7], [8]. Here all application load and processing is entirely on the cloud and accessed via the mobile device.

To some extent, a number of solutions have been developed to solve the issues with resource scarcity on smartphones. These include architectures such as phone2cloud [9], MAUI [10], clone cloud [11], and ECOS system [12]. Researchers have also developed algorithms towards the same cause such as the phone to cloud algorithm, the ALL algorithm and K-step algorithm. These algorithms have enabled the smartphone to make use of the resourceful servers and clouds through computational offloading. However computational offloading has been noticed to have a number of issues and thus does not solve the problem efficiently.

This paper discusses the problem of resource hunger of mobile accessed games, previous attempts to it and proposes efficient ways of attending to such applications. This survey arguably provides an overview of mobile application environments on mobile phones, computational resources often consumed, algorithms for these game applications and their inefficiencies when run locally. This paper is further organized as follows: Section-2 describes a background on mobile smartphones and resource hungry games; Section-3 summarizes resource monitoring tools; Section 4 analyses the existing algorithms; Section-5 discusses the results obtained in the survey and Section-6 proposes an improved approach for resource hungry mobile applications.

2. SMARTPHONE TECHNOLOGY

2.1 Mobile Phone's Storage

Every phone is different, and in most cases one needs to read the specs on the phone to get the details. But, In general, the way Android refers to its memory is: Random Access Memory (RAM), Internal Flash memory and SD Card.

2.1.1 Random Access Memory (RAM)

Just like on any other hand held device, RAM is where applications are executed and where any data of an application it is using at the moment is stored. (When one opens an application, it's read from storage into RAM and then executed.) RAM is not permanent and is routinely cleaned up by the Operating System (OS) as programs are closed. Every OS, be it Windows, OS-X, or Android, handles the details in vastly different ways, but this is how Android OS works [13].

The general rule-of-thumb is "The more RAM you have, the more applications you can run simultaneously". The keyword is "run". The amount of RAM has no bearing on how many applications you can install. It is possible to install 100's of gigabytes of game applications but probably won't be able to run them all at the same time. This means that currently smartphones support games that must be locally installed on them i.e. on their internal storage device in order to run them well. Therefore this can be overcome this by installing games on the cloud but access and play them via the mobile smartphone.

2.1.2 Internal Memory

The internal storage is flash memory, just like a Secure Digital Card (SD Card). It's only built-in and can't be changed. Android divides the internal memory into two partitions:

One partition is marked as read-only by the system. This "read-only" partition is where the Android OS actually lives. Programs and users can't modify this without "rooting" the phone. (This read-only system partition is also referred to as Read Only Memory (ROM), but this is actually a misnomer as the memory is technically read write (R/W) [14].

The other partition of the internal memory is used to store the applications one installs plus their data. This partition is also locked down, but not quite as heavily as the system partition. Each application can modify its own files, but in general, can't access or see other applications. The user doesn't have direct access here, either, without rooting their phone. That’s why backup programs usually require root access in order to back up the application and all of its data.

2.1.3 SD Card

This is the little SD memory card you install in the phone. It's marked as "public" storage by the system. Meaning any one, be it an application or the user, can write, read, delete, and modify any of the files on this card. On stock android, game applications can also be installed here, but only if the developer enables this. The SD Card is used by the user to store anything they want [15].

2.2 Mobile Games

With the advancement and popularity of smartphones, mobile games have become the most used mobile applications. Mobile games are in two major categories i.e. the casual games that are mainly designed to play on feature phones running Symbian operating system and the more advanced resource hungry games that run on other operating systems like Android, windows, etc.

Casual games are very simple to play since they are based on very simple rules, basic techniques, simple strategies and do not require special skills. These games are played in short bursts, during work breaks or, in the case of portable and cell phone games, on public transportation.
Due to their simple characteristics they can be played on the majority of feature cell phones and hence they are immediately available to casual consumer, people who cannot be defined as typical gamers, instead, they play games when they come across them. Tetris, one of the best game sellers in the mobile gaming scenario, is an example of casual game. However with the coming of smartphones, better and advanced games have been developed thus attracting more game lovers such games include candy crash saga, soccer, etc. [16].

2.2.1 Candy Crush Saga mobile game

Candy Crush Saga is among that class of mobile games that defines the "casual" genre. It was developed by king.com [17]; it requires a minimum of 30MB storage space and android OS version 2.2 and above. Candy crush saga contains huge number of puzzles, its easy game play and has challenging variations. The levels are either too hard or too easy and even easier to get addicted to; Candy Crush Saga could easily suck up all your time if you let it. If you want mindless color-matching, then look no further. The following are the candy crash saga system recommendations 800 MHz or higher CPU; 512 MB RAM, Tegra 2, SGX 540, and Mali 400, Adreno 205 or higher, 30MB free space and 2.2+ Android O/S.

2.2.2 Need for speed most wanted for Android phones

Need for Speed, also known by its initials NFS, is a racing mobile game franchise published by Electronic Arts and developed by several studios including EA Black Box, Criterion Games and Ghost Games [18].

The series released its first title, The Need for Speed in 1994. The title comes from a famous quote from the 1986 film Top Gun. All installments of the series consist of racing cars on various tracks, with several titles including police pursuits in races. Since Need for Speed: High Stakes, the series has also integrated car body customization into gameplay. Need for Speed is the most successful racing mobile game series in the world, and one of the most successful mobile game franchises of all time. Over 150 million copies of games in the series have been sold to date [19].

NFS mobile game which belongs to the genre of racing games was developed by EA Swiss Sarl and occupies 2GB of memory space running on android OS version 2.3.3 and above. NFS MostWanted Recommended System Requirements include Dual Core 1 GHz or higher CPU; 1

GB RAM, Tegra 3, SGX 540, Mali 400 or higher, 2 GB free space and 2.3.3+ Android O/S.

2.2.3 Poly Path

Poly Path is a puzzle game of focus and attention [20]. While playing the game you are supposed to guide your little guys down the right path. This game helps you to challenge your multi-tasking skills in this fun puzzle. The current version of the game consumes 7.6 MBs of storage space, and it runs on android version 2.3 and above. Poly path game is one of the games that can be smoothly played on smartphones with constrained resources without raising issues of fast battery drainage processor speeds and shortage of storage.

2.3 Mobile phones

The survey that was carried out considered two kinds of smartphones depending on their specifications. Both phones were using android operating systems one with OS v5.0 (lollipop), quad core 1.5GHz CPU, Adreno 430 GPU, 3GB RAM and 32GB internal memory [21]. The other phone with OS v2.3 (gingerbread) [22], 1.0GHz CPU, powerVR SGX531 GPU, 256MB RAM and 512MB internal memory. The two phone specifications were chosen because they lay in two different categories in-terms of cost and specifications. Both phones were used to perform an analysis on candy crash saga and poly path games considering three parameter i.e. battery consumption processor performance and storage capacity.

2.4 Mobile game application computation

For a user to use the smartphone for gaming one needs to purchase a device with a multi-core processor and additional storage to hold large games. Most smartphones are also equipped with a touchscreen obviating the need for a physical keyboard. Universal Serial Bus (USB) peripherals such as audio headphones and data transfer cables are also available for smartphones [23].

An operating system is the core component of a smartphone; it’s a platform where the necessary drivers needed to maintain a connection between the software and hardware. It consists of a number of layers i.e. a kernel that manages the drivers that manipulate a smartphone's hardware, such as its built-in camera or USB ports, a middleware which consists of software libraries that form a connection to mobile applications. More so, a smartphone has an application execution environment that contains all the Application Programming Interfaces (APIs) for developers to program new mobile applications for the operating system. Finally there’s an application suite that contains core applications which are packaged with the operating system by default. These applications include phone call software, text messaging, menu screens, calendars, and more. A mobile application is software that a user can install on a smartphone to perform a particular task. For example, Android has a GPS application which allows the user to obtain travel directions in real time, or even track the locations of family members from anywhere in the country.

2.5 Smartphones Battery consumption

Smartphones are getting more powerful, but battery life hasn't caught up to power all the apps we're using.

Smartphone applications generally use a small amount of RAM (around 50 MB), and so lots of these applications can run simultaneously. The OS might decide while multitasking to suspend the applications that are not being used at the time, saving RAM and freeing it up for use for other applications. This is why android Phones appear to be so smooth and responsive when the devices it runs on may only have 512 MB of RAM [24].

Games and those that are 3D in particular (candy crush saga and need for speed) [25], can consume huge amounts of RAM storing game graphics, textures, 3D models and sound. While having 512 MB may seem smooth for running basic applications and the operating system, it may not be enough to store game information without resorting to annoying and frequent loading screens in high-end games. Therefore for one to play such games they would require more computational resources that the phone can’t provide.

The research carried out in 2014 [26] shows that Candy Crush Saga a popular game liked by gamers leads to serious battery drain. Verizon lists the addictive game as a "high-risk Android app," and states that an untouched device running the app will drain the battery 3.2 times faster than usual. Therefore the phone would need to be recharge over and over again. In addition, the game ranks 10th on a list of battery-draining Android apps by KS Mobile, the developers of the Battery Doctor app.

Battery life is an important factor in providing excellent user experience for smartphones. These devices are intended to last well over 10 hours in normal operation mode and days, if not weeks, in standby mode on a single battery charge. Understanding power consumption of a device while running an application will help application developer make appropriate software design choices to minimize device power consumption it will also help the smartphone user to determine which application consumes the most power.

3. COMPUTATIONAL RESOURCE MONITORING TOOLS

There are two ways by which power consumption can be measured. The first way is to connect special power measurement hardware on the battery terminals of your mobile device. This is an accurate way to measure the total power consumption of everything running on your device.

However, there are software-based methods for doing the same thing, which are much easier and less expensive. For example, mobile apps like battery stats plus Trepn Profiler, Power Tutor or Little Eye are capable of monitoring power consumption of an individual application, or the total power usage of the device.

Powertutor a power monitoring tool that measures the power and processor speed used by each application on the phone was used to measure how much computational resources each application consumed. Powertutor is an online power estimation system that has been implemented for Android platform smartphones. Power Tutor provides accurate, real-time power consumption estimates for power-intensive hardware components including CPU and LCD display as well as GPS, Wi-Fi, audio, and cellular interfaces.

Smartphone owners can use PowerTutor to determine the power consumption characteristics of competing applications, allowing them to make better informed decisions about which applications to use or buy. Most existing application descriptions and reviews do not mention power consumption. PowerTutor also estimates battery lifespan subject to a particular smartphone owner's actual application usage patterns.

The powerTutor and the two game applications were installed on phones with one of the specifications mention in section 2.3 and statistics were captured by the PowerTutor application clearly indicating the battery usage, processor speed and storage use of each single game application. Table 1 below shows the results obtained by powertutor and details of the table are shown by the figures described by in section 5:

Table 1 Resource consumption of mobile games

Game	Time	Storage consumption	CPU Speed	Battery usage
Candy Crush Saga	8min 33sec	50mbs	123.4Hz	1049mW
Poly path	6min 47sec	6.7mbs	460Hz	2057mW

4. EXISTING ALGORITHMS

4.1 Offloading Decision Making Algorithm

Offloading decision making algorithm is the algorithm that was implemented in the phone2cloud architecture. The algorithm takes into account four parameters i.e. average execution time users delay threshold, local power consumption and remote power consumption. The algorithm used the help of the phone2cloud components like the execution time predictor, resource monitor, bandwidth monitor and offloading proxy to perform its role.

The offloading decision making algorithm goes through 5 steps before offloading an application. The algorithm first calculates the average execution time of an application running locally predicted by the execution time predictor. Then it compares the average execution time and the users delay tolerance threshold and if the delay time is less than the execution time then offloading takes place. Otherwise it calculates the power consumption of locally running an application and calculates the power consumption of remotely running an application using the previous application logs. There after the algorithm performs a comparison between power consumption on the mobile client and power consumption on the cloud. If the power consumption on the cloud is greater than the power consumption on the mobile client then the application is run locally else it offloads and then the algorithm stops.

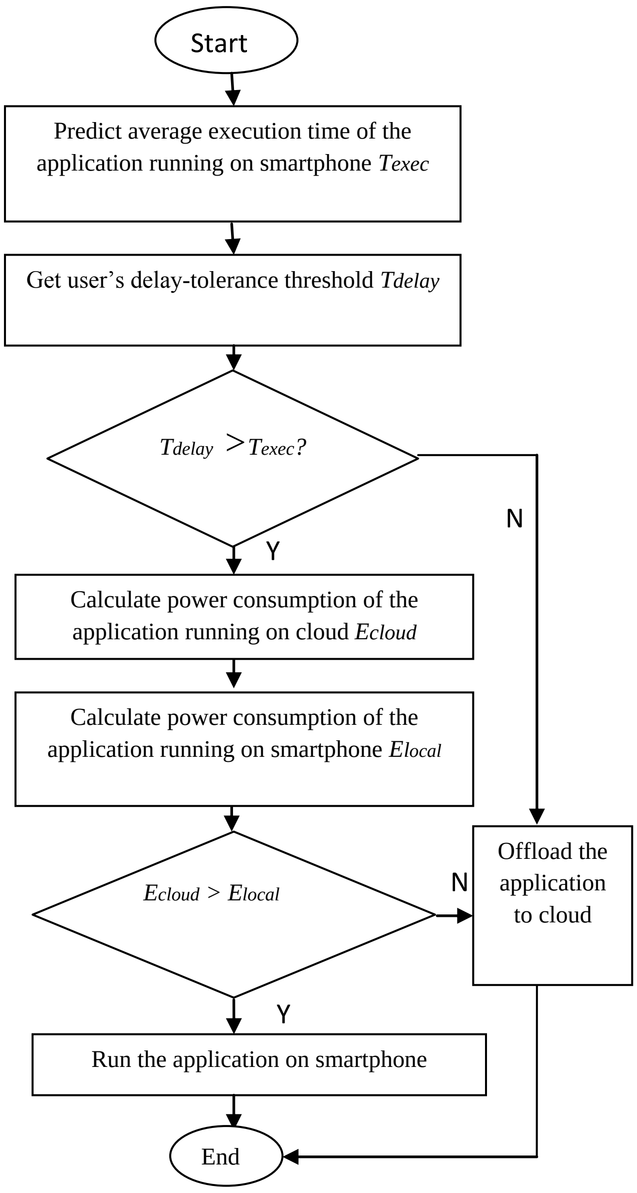

Start
Predict average execution time of the application running on smartphone *Texec*
Get user's delay-tolerance threshold *Tdelay*
Tdelay > *Texec*?
N
Y
Calculate power consumption of the application running on cloud *Ecloud*
Calculate power consumption of the application running on smartphone *Elocal*
Ecloud > *Elocal*
N
Offload the application to cloud
Y
Run the application on smartphone
End

Figure 1 Flow diagram for an offloading decision making algorithm [9]

4.1.1 Pros and cons for offloading decision making algorithm

The algorithm was able to save the mobile clients resources in terms of saving energy, reducing battery consumption and reserving memory storage. The algorithm also improved the applications performance by reducing the latency rate and also improved user's experience of smartphones. However this algorithm is not completely automatic as it requires the user to manually input his/her delay tolerance threshold.

4.2 The ALL and K-step algorithms

The ALL and K-step algorithms were implemented by Giurgiu [27], the algorithms take into account five parameters that is the type of the bundle (movable/non-movable), memory (memory consumption of each bundle on a mobile device), code size (size of the compiled of a bundle), in (the amount of data that a bundle takes in as input from another bundle), out (the amount of data that a bundle sends out as output to another bundle). The algorithms also used the help of AlfredO and OSGi frameworks that helped in application module decomposition, distribution, generation of local and remote proxies and generation of a consumption graph.

The ALL algorithm aims at determining the global optimum cut and performs the optimization offline. The algorithm undergoes three steps before it finally completes all the process. The algorithm generates all valid configurations on the consumption graph and if BundleA and BundleD belong to Bundle Client, BundleA and BundleD are not connected through a direct edge, then all bundles between BundleA and BundleD belong to Bundle Client. Then after identifying the bundles that belong to bundle client, it chooses the bundles that satisfy the phones constraints. Lastly the algorithm evaluates the objective function and chooses the one providing its maximum and minimum value and the algorithm stops.

The K-step algorithm on the other hand aims at determining the local optimum cut and optimization is performed online. Unlike the ALL algorithm, K-step computes the best configuration at each step of the consumption graph. At each node it calculates the resource needed for each bundle and tests them against the resources constraints of the mobile client. At each step it determines the local optimum cut and compares the previous node to the current one and if the current favors the client resource better than the previous then it drops the previous and takes on the current node. The algorithm also maintains a queue of all nodes on the consumption graph that are not yet acquired or tested. Once the algorithm has tested all the bundles, it then determines the graph cut and offloading takes place. Before the algorithms were run, measurements were carried out on time to fetch, install and start bundles and also generate the R-OSGi proxies to enable the interaction between the server and the client.

4.2.1 Pros and cons of ALL and K-step algorithms

The algorithms were able to save memory, battery consumption and storage space on the client. The algorithms also support heterogeneous client side environments.

However to achieve all this modularization was required for each application and since it was performed manually it would take more time. Also the modularization at the service logic level may involve changes in the user interface. More so the decision of the application bundle distribution is server dependent and that's not favorable in terms of server scalability and smartphone dynamic resource requirements

5. RESULTS

To measure the game resource utilization an experiment was run using powertutor a resource monitoring tool for android smartphones. Games were run on an android phone of OS v2.3, 1.0GHz CPU, powerVR SGX531 GPU, 256MB RAM and 512MB internal memory. Experiments were run using PolyPath and candy crush games because they are the most commonly played and loved games by smartphone users as they are found to be so addictive and resource hungry. The games were found to have used the phones resources differently as shown in table 1 previously.

Figures 2, 3, and 4 show how the PolyPath game application consumed the phones resources in terms of the energy consumed by the CPU. Figure 2 shows that PolyPath is the one that consumed the most energy with 51.4 percent, figure 3 is a pie chart showing the energy usage over the last five minutes indicating that the game consumed more energy with the light display due to its high definition graphic features, and figure 4 shows the statistics of energy consumption.

Figures 5, 6, 7 show how candy crush saga game application consumed the phones resources. Figure 5 shows that candy crush saga is ranked second with 25.5 percent after being played for 8 minute and 33 seconds, figure 6 is the chart showing the statistics of energy usage by the game over time, figure 7 is a pie chart displaying the energy usage overtime indicating that the game utilized more energy with the light display because of its high graphic features. Below are graphs showing the power usage of both candy crush saga game and poly path:

Poly path Game Statistics

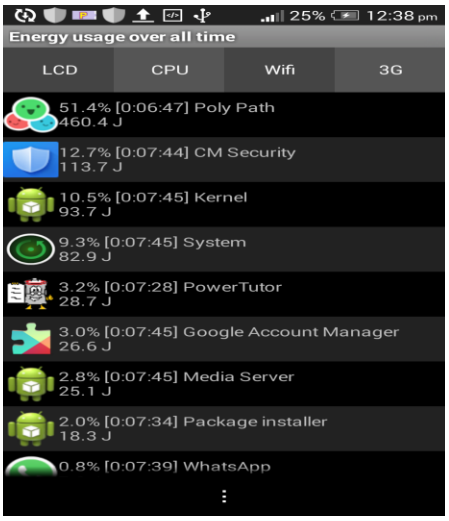

25% 12:38 pm
Energy usage over all time
LCD CPU Wifi 3G
51.4% [0:06:47] Poly Path
460.4 J
12.7% [0:07:44] CM Security
113.7 J
10.5% [0:07:45] Kernel
93.7 J
9.3% [0:07:45] System
82.9 J
3.2% [0:07:28] PowerTutor
28.7 J
3.0% [0:07:45] Google Account Manager
26.6 J
2.8% [0:07:45] Media Server
25.1 J
2.0% [0:07:34] Package installer
18.3 J
0.8% [0:07:39] WhatsApp

Figure 2 Energy usage by mobile applications over time

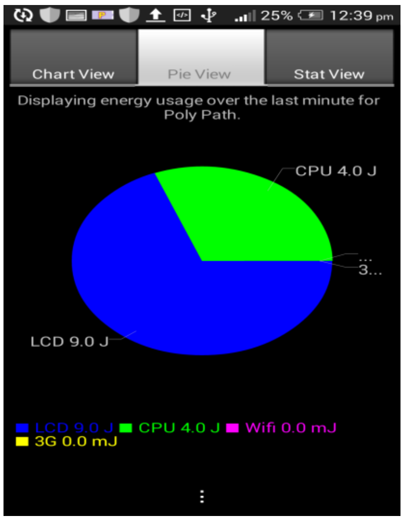


Figure 3 chart displaying the energy usage over time

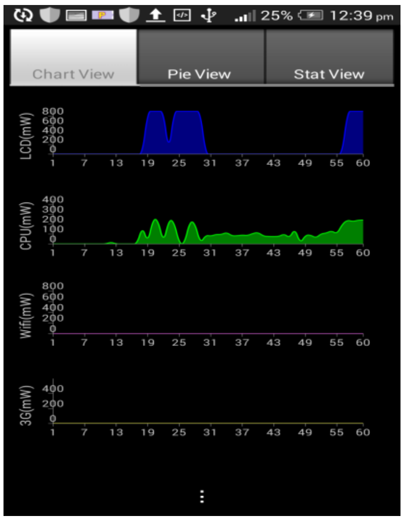
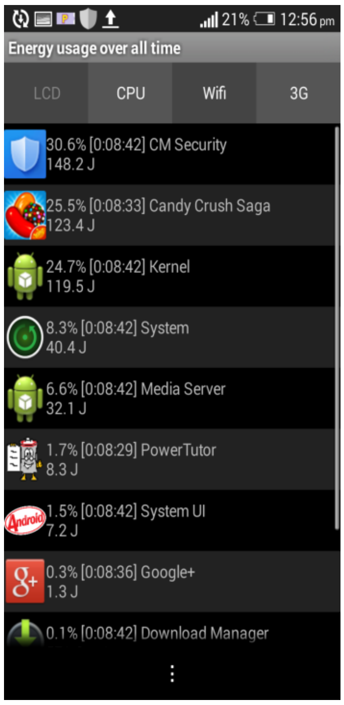


Figure 4 pie chart displaying the energy usage over time

Candycrush Mobile Game Statistics



Energy usage over all time

30.6% [0:08:42] CM Security
148.2 J

25.5% [0:08:33] Candy Crush Saga
123.4 J

24.7% [0:08:42] Kernel
119.5 J

8.3% [0:08:42] System
40.4 J

6.6% [0:08:42] Media Server
32.1 J

1.7% [0:08:29] PowerTutor
8.3 J

1.5% [0:08:42] System UI
7.2 J

0.3% [0:08:36] Google+
1.3 J

0.1% [0:08:42] Download Manager

Figure 5 Energy usage by mobile applications over all time

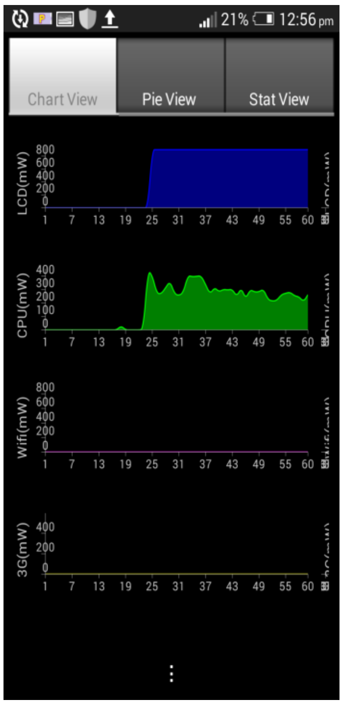
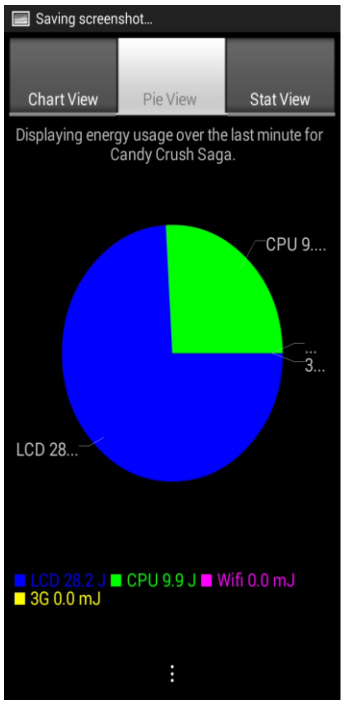


Figure 6 chart displaying the energy usage over time



Displaying energy usage over the last minute for Candy Crush Saga.

Figure 7 pie chart displaying the energy usage over time

Table 2 Resource conservation by the algorithms

Algorithm	Save a lot of Battery	Save Memory	Improve computational Speed	Automatic	Dynamic	Static
Offloading decision making algorithm	yes	yes	no	no	yes	no
ALL algorithm	yes	yes	yes	no	no	yes
K-step algorithm	yes	yes	yes	no	yes	no

It was noticed that running the game application locally consumes more of the phones resources and that the games are not run efficiently and smoothly. Hence having the game applications stored on a cloud or external server would help to increase the game applications processing speed and also save the smartphones RAM to run and process the phones basic operation with ease and smoothness. More so, the survey indicates that running mobile cloud games is more energy efficient than native mobile games. The experiments showed that mobile cloud games reduce the CPU utilization by half, and save energy by 30 percent.

6. CONCLUSION

A number of games are increasingly being developed at a high rate given the continuous advancement in technology. Section 2 discussed a few of the games that are resource intensive, and it has been noticed that as smartphone technology is advancing, little has been put in consideration about the limited resources of these phones yet their functionalities are improved every now and then. Section 4 also discussed the algorithms that have been developed to solve the issues with the resource limitations however they are not effectively and efficiently solving the problems. This survey shows the need to develop better and advanced efficient algorithms that can handle the issues automatically with the support of mobile cloud computing technology. Therefore the researchers propose to design and develop algorithms for accessing resource hungry games on a cloud via mobile smartphones where all the application computation can be on the cloud.

7. REFERENCES

[1] N. Vallina-Rodriguez and J. Crowcroft, "Energy management techniques in modern mobile handsets," Communications Surveys & Tutorials, IEEE, vol. 15, pp. 179-198, 2013.

[2] A. a. report. (2015, 05.10). Avg study reveals top 10 apps that ruin your smartphones mojo. Available: http://now.avg.com/top-10-apps-that-ruin-your-smartphone-mojo

[3] I. Ha, Y. Yoon, and M. Choi, "Determinants of adoption of mobile games under mobile broadband wireless access environment," Information & Management, vol. 44, pp. 276-286, 2007.

[4] P. Mell and T. Grance, "The NIST definition of cloud computing," 2011.

[5] 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, vol. 13, pp. 1587-1611, 2013.

[6] L. Liu, R. Moulic, and D. Shea, "Cloud service portal for mobile device management," in e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on, 2010, pp. 474-478.

[7] 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, 2009, pp. 627-634.

[8] C. Vecchiola, X. Chu, and R. Buyya, "Aneka: a software platform for .NET-based cloud computing," High Speed and Large Scale Scientific Computing, vol. 18, pp. 267-295, 2009.

[9] F. Xia, F. Ding, J. Li, X. Kong, L. T. Yang, and J. Ma, "Phone2Cloud: Exploiting computation offloading for energy saving on smartphones in mobile cloud computing," Information Systems Frontiers, vol. 16, pp. 95-111, 2014.

[10] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, et al., "MAUI: making smartphones last longer with code offload," in Proceedings of the 8th international conference on Mobile systems, applications, and services, 2010, pp. 49-62.

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

[12] A. Gember, C. Dragga, and A. Akella, "ECOS: Practical Mobile Application Offloading for Enterprises," in Hot-ICE, 2012.

[13] A. Carroll and G. Heiser, "An Analysis of Power Consumption in a Smartphone," in USENIX annual technical conference, 2010.

[14] S. Mohtasebi, A. Dehghantanha, and H. G. Broujerdi, "Smartphone forensics: a case study with Nokia E5-00 mobile phone," International Journal of Digital Information and Wireless Communications (IJDIWC), vol. 1, pp. 651-655, 2011.

[15] H. Kim, N. Agrawal, and C. Ungureanu, "Revisiting storage for smartphones," ACM Transactions on Storage (TOS), vol. 8, p. 14, 2012.

[16] M. Furini, "Mobile Games: What to expect in the near Future," in GAMEON, 2007, pp. 93-95.

[17] K. Alha, E. Koskinen, J. Paavilainen, J. Hamari, and J. Kinnunen, "Free-to-play games: Professionals' perspectives," Proceedings of Nordic Digra, vol. 2014, 2014.

[18] V. Gaikar, "Top 10 Games for Samsung Galaxy S4," 2013.

[19] D. Clevert, "Top 5 Products."

[20] S. Hitchon. (2014, 03.16). Poly path mobile game.

[21] P. Nual. (2011, 03.16). How to extend your samsung galaxy s2s battery life

[22] A. Okafor. (2015, 03.18). Techno phone specifications. Available: http://phones.tekinuzu.com/tecno.html

[23] T. L. Rakestraw, R. V. Eunni, and R. R. Kasuganti, "The mobile apps industry: A case study," Journal of Business Cases and Applications, vol. 9, p. 1, 2013.

[24] G. P. Perrucci, F. H. Fitzek, and J. Widmer, "Survey on energy consumption entities on the smartphone platform," in Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd, 2011, pp. 1-6.

[25] D. Ferreira, A. K. Dey, and V. Kostakos, "Understanding human-smartphone concerns: a study of battery life," in Pervasive computing, ed: Springer, 2011, pp. 19-33.

[26] K. Singh. (2014, 23 march). These 8 popular games totally kill your phone battery Available: http://mashable.com/2014/10/01/games-battery-draining/#sSGZx5wzf8qU

[27] I. Giurgiu, O. Riva, D. Juric, I. Krivulev, and G. Alonso, "Calling the cloud: enabling mobile phones as interfaces to cloud applications," in Middleware 2009, ed: Springer, 2009, pp. 83-102.