A Prototype for Distributed Computing Platform using Smartphones

Jeffrey Wagner
wagnejef@mail.gvsu.edu

Xiang Cao
caox@gvsu.edu

School of Computing and Information Systems
Grand Valley State University
Allendale, Michigan 49401, USA

Abstract

Distributed computing usually provides a mechanism for multiple computers to participate in computing tasks. In distributed computing, a large computing job can be divided and sent to many computers, which communicate and coordinate via networks. In recent years, mobile devices like smartphones continue to grow in power and number, so that their combined computing capacity has increased as well. However, much of this computing capacity is wasted, because smartphones sit idle from time to time throughout the day and while charging at night. In this paper, we explore the possibility of harnessing that otherwise unused computing power of smartphones through the implementation of a mobile-based distributed computing platform designed to tackle large computing tasks. To demonstrate the capabilities of the system, we use identification of genes associated with the development of renal cancer as the computing task. Our prototype is set to identify such genes mainly using only the computing power of smartphones for statistical analysis. Performance evaluation shows our prototype design is feasible and promising compared to a centralized system running on a desktop computer.

Keywords: Distributed Computing, Prototype, Mobile Device, Smartphone, Gene Identification

1. INTRODUCTION

Distributed computing has been very prevalent in recent years. Instead of providing services in a centralized solution, distributed computing often involves multiple computers to participate in computing tasks. There are many applications in this broad idea of distributed/decentralized computing, such as Peer-to-Peer (P2P), Blockchain, Internet of Things (IoT).

In distributed computing, a large computing intensive job can be divided and assigned to many computers, which communicate and coordinate via networks. Each computer can work on a small part of the original job respectively. With multiple computing devices working together, many distributed/decentralized computing applications are expected to produce results more efficiently.

Admittedly, there are some research work such as (Appuswamy, Gkantsidis, & Narayanan, 2013) and (Karanasos, Rao, Curino, Douglas, Chaliparambil, Fumarola, Heddaya, Ramakrishnan, & Sakalanaga, 2015), which discuss the advantages and disadvantages of distributed computing paradigm, compared with more centralized approaches. Also, using idle computing power is a way to increase utilizations and efficiently explore the available computing resources. However, different from other work, we focus on the idle computing power of smartphones. In this paper, we look at the

distributed computing from a different perspective - exploring the unused computing resource of smartphones.

Recently, mobile devices like smartphones have more and more improved computing power, so that each of them can be viewed as a computing device. When Apple released the iPhone in 2007, it started a technological revolution that cellphone continues to grow in popularity and power today. Also, the number of smartphones has grown tremendously. As 2019, 81% of Americans owned a smartphone (Pew Research Center, 2019) and those smartphones have gone from iPods that can make phone calls to very capable computers that fit into our pockets. Hence, the combined computing capacity of smartphones have become a significant resource.

With so many powerful smartphones always on and often sitting unused (smartphones are idle from time to time throughout the day and while charging at night), a lot of this computing resource is wasted. In this paper, inspired by the idea of "Citizen Science" (Wikipedia – Citizen Science, n.d.) and "Crowdsourcing" (Wikipedia – Crowdsourcing, n.d.), we explore the possibility of harnessing that otherwise unused computing power of smartphones.

We implement a mobile-based distributed computing platform designed to tackle large computing tasks. We develop a prototype consisting of smartphones as the main computing resource. To demonstrate the effectiveness and efficiency of our system, we use identification of genes associated with the development of renal cancer as the computing task. Our prototype is set to identify such genes mainly using only the computing power of smartphones to run statistical analysis tasks. Compared to a centralized system running on a desktop computer, performance evaluation shows our prototype design is feasible and promising.

The rest of this paper is organized as follows. Section 2 is the literature review. In Section 3, we introduce the prototype architecture. We show our implementation in Section 4. In Section 5, we show the performance evaluation and discuss our results. Finally, we conclude our paper and present the future work in Section 6.

2. LITERATURE REVIEW

Citizen science (Wikipedia – Citizen Science, n.d.) has been popular in recent years. In citizen science, amateur scientists have chances to participate in scientific research projects, by offering their data, knowledge, experience, equipment, devices and so on. Similarly, crowdsourcing (Wikipedia – Crowdsourcing, n.d.) is a model that assigns and divides tasks among participants to obtain a combined result. Nowadays, Internet is widely utilized in citizen science and crowdsourcing to involve participants. Related to this paper, we believe if some people could contribute the idle time of their smartphones, extra computing resources would be available.

There are many distributed computing projects (Wikipedia – List of Distributed Computing Projects, n.d.) that allow the public to contribute their spare computing power. For example, the Folding@Home project (Folding@Home, n.d.) allows users with home computers, particularly those with powerful GPUs, to contribute towards disease research by allowing their machines to be used as part of Folding@Home's distributed system, which processes immense datasets related to protein folding. SETI@Home (SETI@Home, n.d.) is a volunteering computing project to analyze data for searching intelligent life in the universe. Users connected to Internet can participate in this project by downloading and analyzing radio signal data using their personal computers.

Some distributed computing projects involve smartphone usages. For example, SPOTTERON (SPOTTERON, n.d.) is a platform offering solutions for citizen science and volunteering monitoring projects. Customized smartphone Apps are provided in the platform. iNaturalist (iNaturalist, n.d.) allows users to observe and share biodiversity on the earth using smartphone Apps. In (Graham, Henderson, Schloss, 2011), three examples of involving citizen scientists in research using mobile phones have been discussed.

Research work in (Arslan, Singh, Singh, Madhyastha, Sundaresan, & Krishnamurthy, 2012), (Duan, Kubo, Sugiyama, Huang, Hasegawa, & Walrand, 2014), and (Remédios, Teófilo, Paulino, & Lourenço, 2015) have studied smartphone issues in distributed computing. In (Arslan et al., 2012), several aspects of smartphones have been investigated, including profiling battery charging behaviors, task migration and task scheduling on smartphones. In (Duan et al., 2014), different incentive mechanisms about motivating smartphone users to participate in data acquisition and distributed computing are analyzed and proposed. Study in (Remédios et al., 2015) describes a preview of a distributed mobile system to process locally

generated data in a network of smartphones without infrastructure support.

In this paper, we present a hands-on experimental study that shows the feasibility and effectiveness of distributed computing using smartphones.

3. PROTOTYPE ARCHITECTURE

Our prototype consists of two components, a simple (central) server and multiple (client) smartphones, as shown in Figure 1.

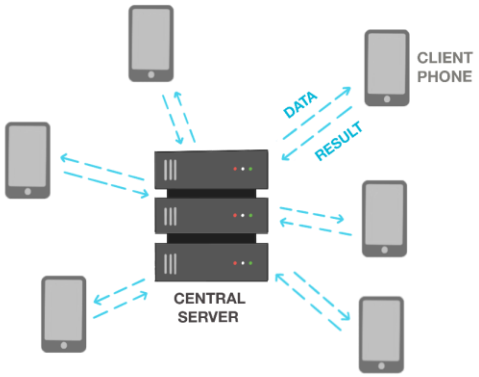

Figure 1. Prototype Architecture

The (central) server in our prototype does not participate in running the actual computing tasks (i.e., statistical analysis of gene identification). Instead, it divides the large computing task, communicates with all the (client) smartphones and coordinates them. The (central) server sends the partitioned data to multiple (client) smartphones, waits for the results returned by them, and finally assembles results into a meaningful output.

The (client) smartphones in our prototype receive the data from the server, do the actual computing for statistical analysis of gene identification, and send the results back to the server.

The detail of the implementation is presented in the next section.

4. IMPLEMENTATION

Server

The server divides the entire input file into many pieces and sends them one by one to different client smartphones for processing. The server makes use of multithreading and concurrency in its design. There is a command line interface that accepts user inputs and acts accordingly, a thread that listens for new clients (smartphones) connecting and provides them with TCP socket connections, and threads that interact with each of the client smartphones to send data and receive results. The server program runs in a regular desktop machine in Java using JetBrains IDEs, IntelliJ.

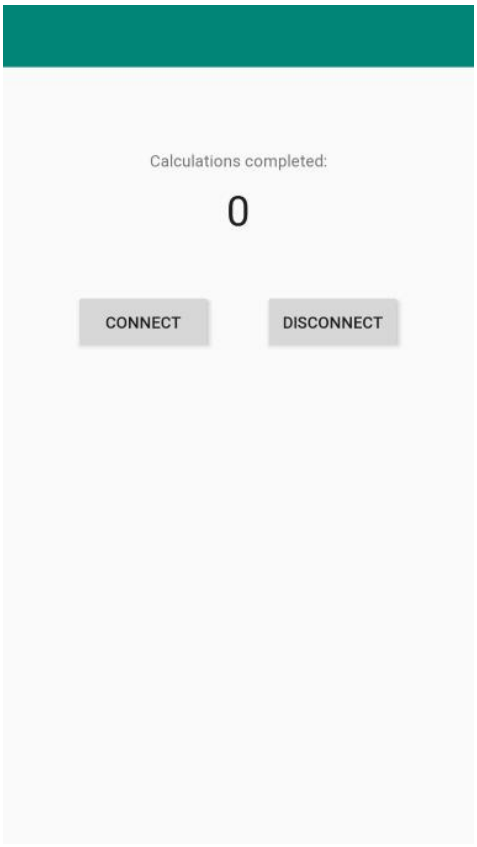

Figure 2. Mobile App Interface

Client Smartphones

Client smartphones can connect to the server for data and return the results. Each smartphone runs an Android mobile App implemented in Java using Android Studio. The App consists of a simple user interface as shown in Figure 2, along with a back end for handling data processing. When the user of the smartphone wants to contribute some processing power, the App can be launched and the "CONNECT" button is pressed to join the system. The App will then listen for data coming in and process it on arrival, then return the result back to the server and await the next job. A counter in the interface shows the number of calculations completed. In this example case, it is the number of genes that have been analyzed for their association to renal cancer. Once a gene analysis job is complete, the client smartphones wait on standby and the

server outputs a CSV file with the compiled results.

Statistical Analysis
For statistical analysis jobs run by smartphones, a Cohen's d (d-score) is calculated for each gene using data from 8 renal cancer patients and 52 non-renal cancer patients from the National Cancer Institute's NCI-60 dataset (CellMiner, n.d.). In our experiments, calculating the d-score involves calculating the mean, standard deviation, and t-statistic for the 60-item dataset, then repeatedly shuffling the dataset and recalculating the statistics until a final comparison result in the d-score.

Efficiency
For better efficiency, the server keeps listening for results from the smartphones. As soon as it receives one from a smartphone, the server sends that smartphone another piece of data to continue to work on.

In our experiments and in the real practical system, smartphones have different data processing speed because of their various specifications. Some smartphones process data faster with better CPUs and others run experiments slower. If there are only a few pieces of input data left and the slower smartphones are still processing them, the server will also send these data to the faster smartphones to take advantage of their higher processing speed, preventing them from going idle. The server will use whichever results come back first from slower and faster smartphones to assemble the compiled result.

Robustness
Our prototype is also designed to be robust in its handling of client smartphones. When other smartphones are running their jobs, a new client smartphone can still connect to the server for its task. When a smartphone is disconnected, it will not cause issues. Its work will be reassigned to another available client smartphone. The robustness is tested by intentionally connecting and disconnecting smartphones during trial runs, and no result is corrupted due to that.

5. PERFORMANCE EVALUATION AND DISCUSSION

Devices
In our experiments, we use 5 smartphones to test our platform, as shown in Table 1. The Google Pixel 2 XL was a former flagship smartphone (released in October 2017), but now it is considered mid-range in today's market. The Motorola Moto X4 is a more budget friendly model, and the Blue Advance A4 is a very inexpensive Android smartphone. We choose these smartphones because the Google Pixel 2 XL and Motorola Moto X4 are the ones we have had, and the Blue Advance A4 is the cheapest smartphones we could find. These smartphones run Android systems, so that IOS implementation is not included.

For the desktop machine, its CPU is AMD FX 8350, and the GPU is Nvidia GTX 980.

Device	Processor	Retail Price	Quantity
Google Pixel 2 XL	Snapdragon 835	$175	1
Motorola Moto X4	Cortex A53	$140	1
Blue Advance A4	Cortex A7	$40	3

Table 1. Smartphones for Performance Evaluation

Dataset
The dataset for gene analysis in our experiments is a pre-processed copy of "RNA: Affy HG-U133 Plus 2.0" from the National Cancer Institute's NCI-60 dataset (CellMiner, n.d.). The size of the dataset is about 1.8MB. Each gene is shuffled 10,000 times before the final result comes out.

Performance Metric
The experiments are measured by their processing time as the performance metric. In our experiments using smartphones, the processing time is the wall-clock time from the server sending out the first piece of data, to the moment when the server receives the last result. For the server-only experiments, the processing time is the time for the desktop machine to finish the entire gene analysis job. For each experiment, we run three times and show the average as the result.

Experiments
(1) Server-only
In the first group of experiments, we run our gene statistical analysis jobs on the desktop machine, exploring its computing capability. We also utilize the desktop machine to simulate the performance of a typical centralized (non-distributed) server solution. This set of experiments is a benchmark compared with the performance of smartphones.

The CPU of the desktop machine is AMD FX 8350, which has 8 threads. In order to test the impact

of concurrency, we run our gene analysis program on the desktop machine using different number of threads. The result is shown in Figure 3.

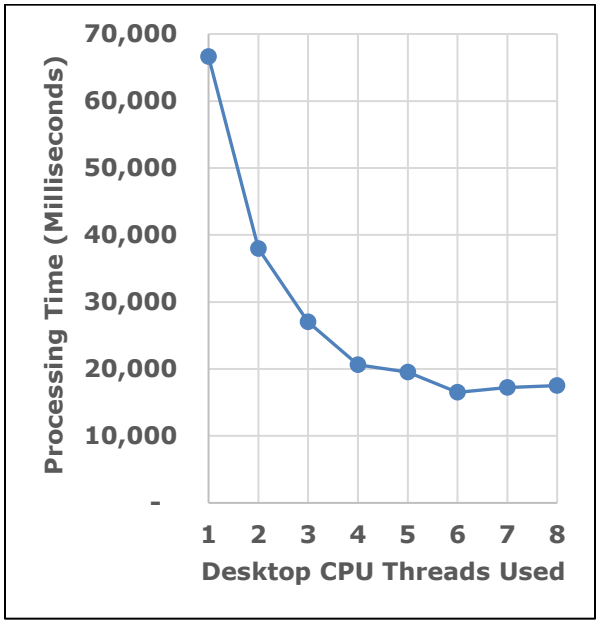

Figure 3. Processing Time vs. Number of Threads

From the result, we can see that when the number of threads used is small, the processing time decreases as the number of threads increases, because of the concurrency. However, the decrease in processing time does not scale exactly with the number of threads used. For example, the 2-thread run is about 1.8 times as fast as the single thread run, and the 3-thread run is about 2.5 times as fast.

When the number of threads reaches a certain value, the performance is little changed. Hence, the concurrency cannot improve the performance unlimitedly.

(2) Smartphones
We conduct experiments to test the computing capabilities of smartphones. We first run the gene identification jobs using the slowest but most affordable smartphone in our device pool - Blue Advance A4, exploring the performance related to the number of smartphones.

Figure 4 shows the result. We can see that the performance is reversely proportional to the number of smartphones, because of the parallelism. The processing time of 2-phone run is almost exactly half as that of the single phone run, and the 3-phone run takes three times faster. This trend of inverse proportionality is more accurate and direct than that of desktop machine's performance with multiple threads. This is because each smartphone runs tasks separately as independent small computers.

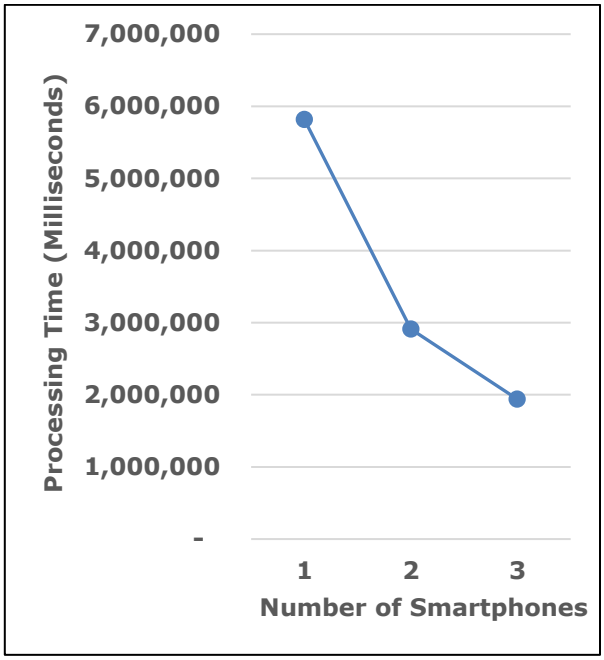

Figure 4. Performance of Blue Advance A4

We then conduct three sets of experiments on 1 Google Pixel 2 XL, 1 Motorola Moto X4, and a group of all smartphones (1 Google Pixel 2 XL, 1 Motorola Moto X4 and 3 Blue Advance A4s) respectively, compared with desktop machine (1 thread). The result is shown in Figure 5.

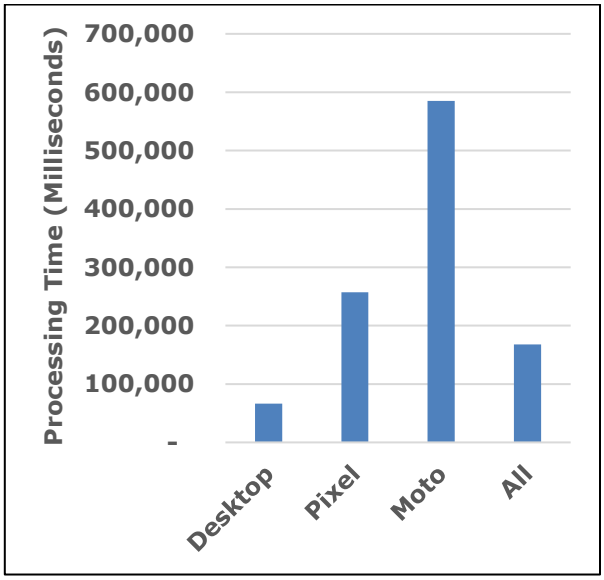

Figure 5. Performance of Different Devices

From Figure 5, we can see that the Google Pixel 2 XL has a better performance compared the Motorola Moto X4. Although the desktop machine still performs the best, a group of all 5 smartphones reduces the processing time by taking advantage of parallelism. Along with the fact of inverse proportionality shown in Figure 4, we believe with more and more smartphones, a group of smartphones will eventually outperform the desktop machine.

Projection

It would be ideal to further test the system with more high-end smartphones by spending thousands of dollars. However, due to our budget and resource limit, we show the performance of a large-scale system by projection based on the fact of inverse proportionality shown in Figure 4 and the actual data from Figure 5. Our extrapolation is shown in Figure 6 in the Appendix.

We extrapolate the performance of up to 36 Google Pixel 2 XL and Motorola Moto X4 smartphones respectively. In Figure 6, the first data points (shown in triangles as markers) of Motorola Moto X4 and Google Pixel 2 XL are actual performance data (the same as shown in Figure 5), while the rest of their data points (shown in circles) are projected according to the pattern demonstrated by the Blue Advance A4 shown in Figure 4.

As shown in Figure 6, it would take 4 Google Pixel 2 XLs or 9 Motorola Moto X4s to outperform a single thread of the desktop machine. It would take 16 Google Pixel 2XLs to outperform the desktop machine's best multithreaded run (6 threads as shown in Figure 3) or 36 Motorola Moto X4s to accomplish the same feat. We can see that, while a single smartphone does not excel at speed compared with the desktop machine, a group of them together can reduce the processing time. Hence, with more and more smartphones, the performance of the system would be better and better.

Discussion

From all the previous results, we can see that parallelism indeed improves the system's performance, so that multiple smartphones provide better results than a single one.

In today's computing intensive world, computing resources are in high demand. More and more servers are being purchased and added into the computing pool. On the other hand, the processing power of smartphones has been growing rapidly. Consumers have continued to demand more performance from their smartphones for over a decade and manufacturers have been happy to provide support. However, much of computing power of smartphones is wasted when they are sitting idle from time to time throughout the day and while charging at night.

With billions of smartphone users worldwide (Statista, 2020), it would be a wise idea to harness that otherwise tremendous unused "almost free" computing power of smartphones. Our results show that with only several or dozens of smartphones, their processing capabilities can outperform a traditional desktop machine.

Like citizen science, if the computing platform intends to expand its processing power, or accelerate its processing speed on a project, it can consider encouraging people to contribute their smartphones' computing power. Some incentives can be given to attract smartphone users to participate. Based on our experimental results, we have showed that it is a feasible and promising solution, at least from the technical perspective.

6. CONCLUSIONS

In this paper, we investigate the possibility of harnessing that otherwise unused computing power of smartphones. We implement a mobile-based distributed computing prototype using smartphones to tackle large computing tasks. Our experimental results show the prototype is technically feasible and promising.

Our idea of this prototype is supported by two key facts: (1) the number of smartphones has been growing continuously along with their computing power, so that their combined computing capacity has increased in a rapid rate; (2) Much of the computing capacity is wasted when smartphones are sitting idle from time to time throughout the day and while charging at night. Hence, it would be ideal if we could efficiently utilize the computing resources of smartphones.

In future work, we plan to run other types of tasks in smartphones and investigate their impacts on the prototype's performance, compared with the desktop machine. We also plan to investigate the impact of task processing on the smartphones themselves, e.g., CPU and memory usage, power consumption of smartphones. Additionally, more smartphones can be involved to further test the feasibility of the system.

7. REFERENCES

Appuswamy, R., Gkantsidis, C., & Narayanan, D. (2013). Scale-up vs Scale-out for Hadoop: Time to rethink? Publishing in ACM *SoCC*, *20*, 1-13.

Arslan, M., Singh, I., Singh, S., Madhyastha, H., Sundaresan, K., & Krishnamurthy, S. (2012). Computing While Charging: Building a Distributed Computing Infrastructure Using Smartphones. Publishing in ACM *CoNEXT*, 193-204.

CellMiner (n.d.). Retrieved from https://discover.nci.nih.gov/cellminer/loadDownload.do

Duan, L., Kubo, T., Sugiyama, K., Huang, J., Hasegawa, T., & Walrand, J. (2014). Motivating Smartphone Collaboration in Data Acquisition and Distributed Computing. Publishing in IEEE *Transactions on Mobile Computing, 13(10)*, 2320-2333.

Folding@Home (n.d.). https://foldingathome.org

Graham, E., Henderson, S., & Schloss A. (2011). Using mobile phones to engage citizen scientists in research. Publishing in AGU *EOS, 92(38), 313-315*.

iNaturalist (n.d.). https://www.inaturalist.org/

Karanasos, K., Rao, S., Curino, C., Douglas, C., Chaliparambil, K., Fumarola, G., Heddaya, S., Ramakrishnan, R., & Sakalanaga, S. (2015). Mercury: Hybrid Centralized and Distributed Scheduling in Large Shared Clusters. Publishing in USENIX *ATC*, 485-497.

Pew Research Center (2019). Mobile Fact Sheet. Retrieved from https://www.pewresearch.org/internet/fact-sheet/mobile/

Remédios, D., Teófilo, A., Paulino, H., & Lourenço, J. (2015). Mobile Device-to-Device Distributed Computing Using Data Sets. (2015). Publishing in EAI *MOBIQUITOUS*, 297-298.

SETI@Home (n.d.). https://setiathome.berkeley.edu

SPOTTERON (n.d.). https://www.spotteron.net/

Statista (2020). Number of smartphone users worldwide from 2016 to 2021. Retrieved from https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

Wikipedia – Citizen Science (n.d.). Retrieved from https://en.wikipedia.org/wiki/Citizen_science

Wikipedia – Crowdsourcing (n.d.). Retrieved from https://en.wikipedia.org/wiki/Crowdsourcing

Wikipedia – List of Distributed Computing Projects (n.d.). Retrieved from https://en.wikipedia.org/wiki/List_of_distributed_computing_projects

ISSN: 2167-1508
v13 n5348

Appendix

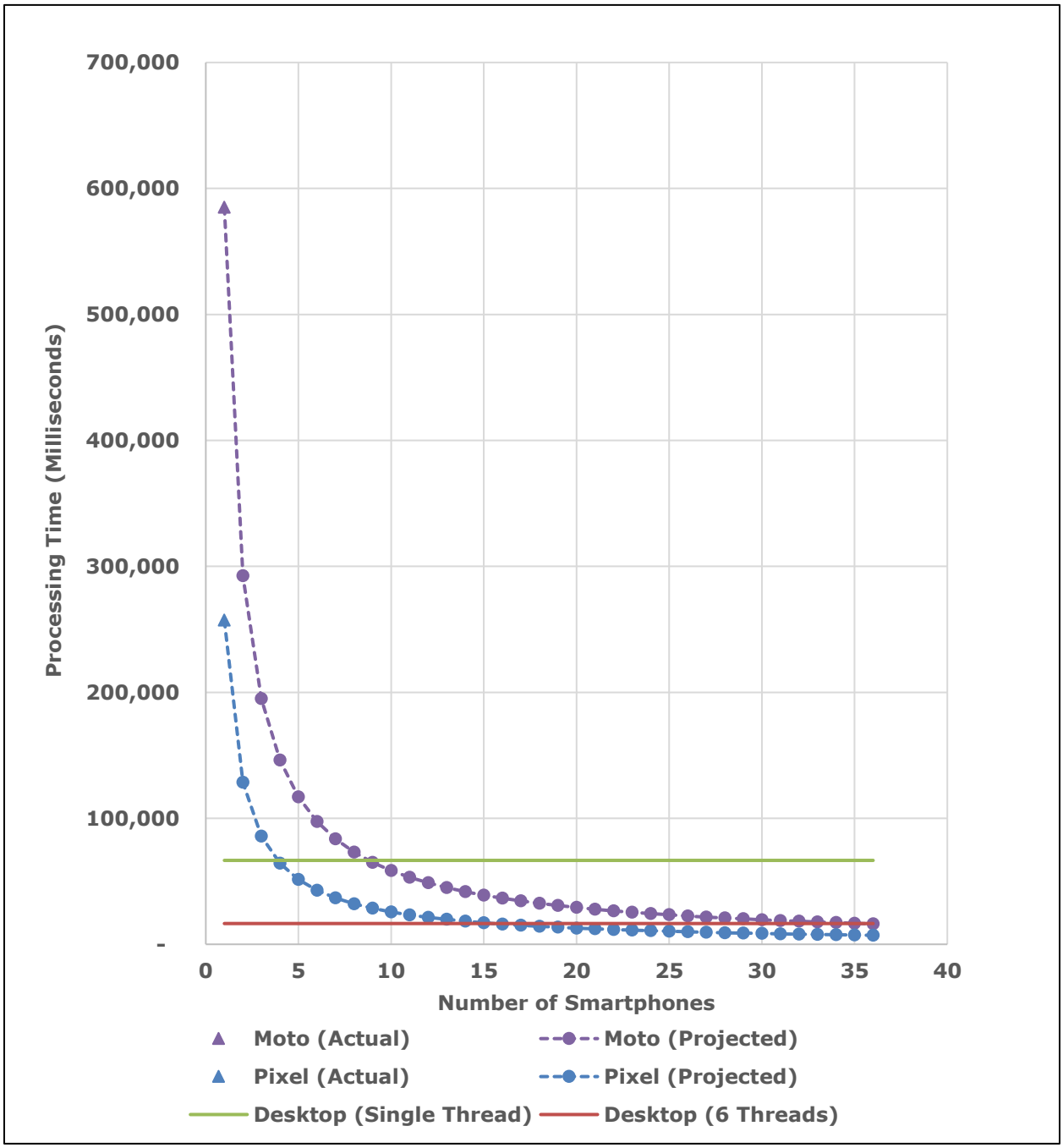

Figure 6. Actual and Projected Performance