Smartphones as Alternative Cloud Computing Engines: Benefits and Trade-offs

Brennan Schaffner*, Jason Sawin*†, and Joseph M. Myre*
*Department of Computer and Information Sciences, University of St. Thomas, Saint Paul, MN
†Email: jason.sawin@stthomas.edu

***Abstract*—Over the past decade, both the popularity and computational power of smartphones have grown at fantastic rates. These devices now represent a wealth of untapped computational power. There have been efforts to incorporate smartphones into volunteer grid frameworks as engines of computing in a manner similar to desktop computers. However, it is not clear what the energy implications of such an integration are. Phones take less energy to create and use less power while running than desktops, but they take significantly longer to complete a task.**

In this paper, we present an initial investigation of the trade-offs of integrating smartphones into the cloud. We created a system to generate computation loads on both smartphones and desktops. In an empirical study, we monitored the energy consumption of both systems under similar computational loads. The results of our study show that the desktop could finish our task 10 to over 30 $\times$ faster than the smartphones. However, on specific benchmarks, we observed that the desktop consumed up to 2.26 $\times$ more energy than the most efficient phone.

***Index Terms*—cloud computing engines; smartphone; energy consumption; volunteer resources;**

I. Introduction

Corporations like Google and Amazon serve hundreds of millions of customers every year. To meet this demand, they create massive server farms. In fact, the construction of new data centers has been estimated to cost 20 billion dollars a year [1]. In addition to being expensive to create, data centers require significant resources to run. In the United States, data centers account for about 2% of the countries total energy consumption [2]. Though these numbers may be startling, the cloud services provided by these companies might actually represent a net gain for the environment. Since the clients of these servers are sharing an optimized infrastructure and not developing discrete systems, energy is saved, CO_2 emissions are reduced, and less e-waste is produced [3]. A significant portion of these savings is because fewer new resources are being created to meet demand. Greater energy savings can be realized by making even better use of the worlds existing systems.

One approach to better leverage existing infrastructures is the creation of volunteer distributed computing frameworks. These frameworks rely on the public to donate the unused processing power, internet connectivity, and storage of their personal computing devices. The Berkeley Open Infrastructure for Network Computing (BOINC) [4], released in 2002, is currently one of the most popular volunteer frameworks. BOINC enables computer owners to contribute to major computational projects. For example, the SETI@HOME project searches for extra-terrestrial life by using spare desktop cycles from around the world to analyze radio signals [5]. BOINC currently has a network of over 800,000 volunteer computers and is regularly processing over 15 PetaFLOPS [6]. This computational power is achieved by using computing cycles that were most likely going to be wasted.

Originally, BOINC was designed for volunteers to donate time on their desktop computers. At the time desktops were the most ubiquitous computing resource. However, smartphones are quickly becoming the most popular electronic consumer good. The Deloitte Consumer Review's Digital Predictions 2018 estimates that 90 percent of adults in the UK will have a smartphone by 2020 [7]. While the popularity of smartphones has dramatically increased, so has their computing power and capabilities. For example, 2018's Samsung Galaxy S9+ showcases three cameras, 6GB of RAM, a 64-bit octa-core processor with cores clocking at 2.8GHz, a Quad HD+ Super AMOLED display, and an extensive list of sensors.

The computational power of smartphones could be a useful supplement to cloud computing. There are considerable periods of time when the average user is not actively engaging their phone [8]. This downtime could be donated for computation. Adding mobile devices to existing volunteer framework would require only slight software modifications. However, there are limitations when incorporating mobile devices into the world of cloud computing. First, smartphones have a significant reliance on battery. Unlike desktops which are continually plugged in, smartphones stop functioning if their battery is depleted. Second, their connectivity is often intermittent and expensive due to their portability. Lastly, due to their small footprint, smartphones have limited CPUs, storage, and RAM compared to their larger desktop counterparts.

BOINC [9] and HTCs Power to Give volunteer frameworks [10] have recognized the untapped computational power of smartphones and have created methods to incorporate them into their systems. To circumvent some of the limitations of mobile devices, they only use a phone when it is plugged in and connected to WIFI. Their scheduling algorithms anticipate that a task will take longer to complete on the smaller device than on a desktop.

Though incorporating smartphones into the cloud as computing engines makes more thorough use of the world's existing computation infrastructure, the overall energy implications of such an addition are unclear. From a resource creation perspective, it takes less embodied energy (emergy) to make a

978-1-5386-7503-8/18/$31.00 ©2018 IEEE
DOI 10.1109/FiCloud.2018.00043

smartphone than a desktop. It is estimated to take about 1 GJ to create a smartphone, whereas desktops are expected to require approximately 7.5 GJ [11]. Moreover, a smartphone uses much less power than a desktop when working on a computational task. However, it takes a smartphone much longer than a desktop of the same generation to complete the task.

In the paper, we present an initial exploration of the energy trade-offs of running specific tasks on a smartphone versus running them on a desktop. We created a lightweight Andriod application which can introduce variable workloads to computationally expensive benchmark subroutines. We deployed our application to a Moto x^4 and Moto g^5 smartphone. We monitored the phones' energy consumption as they executed three benchmark subroutines under varying loads. We then compared the phones' results to those of a Hewlett-Packard EliteDesk 800 G1 executing the same benchmarks.

The remainder of the paper is organized as follows. In Section II, we provide an overview of contemporary smartphones and previous work related to our own. In Section III, we describe our benchmarks, smartphones, desktop, applications, and energy monitoring system. In Section IV, we discuss the experimental design and the results of our empirical study. We conclude and present future work in Section V.

II. Background and Related Works

In this section, we briefly describe the background and the most relevant prior works related to smartphones' energy consumption and incorporation into the cloud.

A. Smartphones' CPU and Energy Consumption

Current smartphones include a wide array of features and sensors, including GPS Navigation Systems, Accelerometers, WIFI, Magnetometer, and many more. Each of these systems can affect the amount of energy consumed. For this paper, we only consider the energy used to complete specific computational tasks. These tasks mostly rely on the phone's CPU.

All mobile applications are reliant on the CPU, thus it has a significant impact on a phones usability and efficiency. The CPU is also one of the primary consumers of the phone's power [12], [13]. It has even been shown that there is a direct correlation between a CPU's frequency and its energy consumption [14]. The higher the CPU frequency, the more energy that is required.

One way to increase computational capacity without incurring a linear increase in energy consumption is to introduce multiple cores. A significant portion of a CPU's power draw comes from changing the state of logic gates (dynamic power consumption). Dynamic power consumption is roughly proportional to the clock rate and the square of the CPU's voltage [15]. Thus, if a single core running at frequency x takes y time to complete a task, it may be possible to complete the same task in y time on two cores each having frequencies of $x/2$. Given that dynamic power consumption decreases with the square of the voltage and lower frequencies require lower voltage, it is possible that the two-core system uses half the energy of the single core to complete the task. In 2011 LG Optimus 2x was the first mobile phone to use a dual-core processor [16]. Now, dual-core smartphones are commonplace with many higher-end phones having hexa-core or octa-core processors.

To further mitigate the increasing energy demands of high-end CPUs, some smartphones are incorporating a heterogeneous (asymmetric) multi-core architecture. These architectures pair processors designed to maximize computational performance with processors that have a lower, more energy efficient clock rate. Both types of processors use the same instruction-set architecture (ISA). This design allows a phone's OS to schedule computationally expensive procedures like certain video games to the high-end cores and lower-processing intensity tasks, such as texting, to the energy efficient core. This architecture can produce significant energy savings [14].

B. Smartphones and the Cloud

As the computational power and the popularity of smartphones have increased, researchers have been exploring ways of maximizing their potential, including methods for incorporating them into the cloud.

In 2011, Harizopoulos et al. introduced MicroCellstore [17], which re-purposed unused smartphones as data-application mini-clusters. The energy analysis of their system showed that smartphones are more energy efficient than alternative low power systems. Their work focused on database systems and single-core smartphones. Our work is evaluating multiple-core smartphones running computationally intensive applications.

Schildt et al. presented a distributed computing framework that integrated mobile devices and computers both running Android [18]. They argue that incorporating smartphones into a companies grid could lead to cost savings, as they are existing hardware that reduces the cost and energy of creating new servers.

Bushing et al. [19] present a study that evaluated the feasibility of adding smartphones to volunteer computing projects. They created a small MPI cluster using Android mobile phones and demonstrated that it could scale reasonably well. Similarly, both Arslan et al. [20], and Kumar and Senthilkumar [21] proposed distributed computer infrastructures that made use of mobile devices in charging or unused states. Kumar and Senthilkumar determined that it would take two to four Android devices to match the performance of a desktop, regarding FLOPs and IPS. Arslan et al. suggested that it might take up to six phones to equal a desktop.

None of the studies above specifically examine the energy being consumed to complete a task and compare the differences between a desktop system and a smartphone.

III. Hardware and Benchmarks

In this section, we briefly describe the hardware and benchmark subroutines that we used in our empirical study.

We chose the Motorola Moto x^4 and the Motorola Moto g^5 smartphones for our experiments (specifications are provided in Table I). Both phones are Motorola models of the year

Component	Moto x^4 Specifications	Moto g^5 Plus Specifications
MODEL	XT1900-1	XT1687
SOC	Qualcomm Snapdragon 630 SDM630	Qualcomm Snapdragon 625 MSM8953
OS	Android 8.0.0 Oreo	Android 7.0 Nougat
CPU	4x 2.2 GHz ARM Cortex-A53, 4x 1.8 GHz ARM Cortex-A53	8x 2.0 GHz ARM Cortex-A53
RAM	3 GB LPDDR4	4 GB LPDDR3
STORAGE	32 GB	64 GB
POSITIONING	GPS, A-GPS, GLONASS	GPS, A-GPS, GLONASS
DISPLAY	LTPS IPS LCD, 1080 $\times$ 1920 pixels, 24bit	LTPS IPS LCD, 1080 $\times$ 1920 pixels, 24bit
GPU	Qualcomm Adreno 508	Qualcomm Adreno 506
WIFI	802.11 a,b,g,n, Dual Band	802.11 a,b,g,n, Dual Band
BATTERY	3000 mAh, Li-Ion	3000 mAh, Li-Ion
RELEASE	October 2017	April 2017

TABLE I
SMARTPHONE SPECIFICATIONS

2017 with Qualcomm Snapdragon processors of 2016. They have identical screen size, resolution, and battery capacity and represent mid-range phones of that year. The differences are in the OS, RAM, and CPU. The Moto x^4 has four ARM cores clocked at 2.2 GHz and four clocked at 1.8 GHz running on Android 8.0 Oreo with 3 GB RAM. The Moto g^5 Plus features all eight cores clocked at 2.0 GHz and runs Android 7.0 Oreo with 4 GB RAM. Essentially the Moto x^4 is a slight upgrade from the Moto g^5 Plus.

While the Moto x^4 has less listed RAM at 3 GB, its LPDDR4 type is a generation ahead of the Moto g^5 Plus's 4GB of LPDDR3. The fourth-generation RAM is designed to increase memory speed and efficiency on mobile devices. The LPDDR4 showcases top speeds of up to double that of the LPDDR3. It also has a low 1.1V supply voltage which is .1V lower than its predecessor. The improved bandwidth and efficiency are a result of its dual-channel architecture. Increasing the number of channels splits the memory and shortens the distance that data needs to travel, which saves energy [22].

Though the phones' cores have different clock rate arrangements, they both use the ARM Cortex-A53 which is one of the earliest microarchitectures to implement ARM Holding's ARMv8-A 64-bit instruction set. At the time of its release in 2012, ARM claimed their Cortex-A53 was the most power-efficient ARM application processor ever and the worlds smallest 64-bit processor [23]. The cores can work independently as processors, but in the case of smartphones, they are combined into multi-core processor configurations. Each core has a separate VFPv4 Floating Point Unit onboard which is an ARM architecture extension responsible for low-cost floating-point computation in accordance with IEEE Standard for Floating-Point Arithmetic (IEEE 754). Some of the ARM Cortex-A53's efficiency may be because it is a superscalar processor, which means it can perform multiple tasks per clock cycle. In addition, the various cores can be put into more energy-efficient states. Each ARM Cortex-A53 core supports four power states. The two simplest states are when all components are fully on or fully off. The other two states feature external power controllers putting certain components into a retention state. When a component is set to a retention state, it uses only enough power to maintain logic and RAM states. The cores' middle power states have either everything on retention or just the VFP and SIMD in retention while the main core state is on [24].

The desktop we used also falls in the middle of the vast range of contemporary machines. The Hewlett-Packard EliteDesk 800 G1 has 4 cores clocked at 3.5 GHz. It runs 64-bit Windows 7 Enterprise on 16 GB RAM. The Intel i5-4690 was released in 2014 and can run up to 4 threads per core. The listed maximum clock speed is 3.9 GHz. The desktop's specifications are provided in Table II.

Component	Hewlett-Packard EliteDesk 800 G1 Specification
OS	Windows 7 Enterprise
CPU	Intel(R) Core(TM) i5-4690 CPU @ 3.50 GHz GHz
Number of Cores	4 cores
Threads per Core	4 threads
RAM	16 GB
System Type	64-bit OS

TABLE II
DESKTOP SPECIFICATIONS

In our study we employed three benchmark subroutines:

- **Matrix Mult.**: The matrix multiplication benchmark burdens the CPU with 10,000 multiplications of two randomly generated 100 degree integer matrices.
- **SHA-1**: The hash benchmark performs the Secure Hash Algorithm 1 (SHA-1) on 5 million random integers [25].
- **Linpack**: The Linpack benchmark solves 5000 systems of 2000 linear equations with random decimal coefficients [26].

To record the energy being consumed by the phones and the desktop during our experiments, we used the Watts-Up Pro Portable Power Meter. It is an in-line power monitor and recording device with $\pm 1.5\%$ accuracy and a 1 Hz sampling rate.

We developed a lightweight load generating applications for both the phones and the desktop. These applications wrap the benchmark subroutines listed above. For a given subroutine, the user could vary the load being generated by selecting the number of the *load threads* to execute. Each *load thread* ran a single instance of the subroutine. The load generating application completes when all threads finish.

IV. Experiments

For our empirical study, we created a baseline state for both the smartphones and the desktop. For the phones, all applications other than our load generator were closed, and airplane mode was activated, thus minimizing spurious activity. The screen brightness was set to maximum. This increased the constant power draw which improves the accuracy of the Watts-Up Meter's measurements. We also only experimented on the phones when they were plugged in and 100% charged, analogous to existing volunteer frameworks using phones only in this state. The desktop was disconnected from the internet, and only the power used by the tower was monitored. Again, all applications apart from the load generator were closed. We monitored the phones' power consumption with the load generating application open but not generating a load for 30 minutes. The average power consumed during this period was considered the baseline power consumption of the device. A similar approach was used to define the desktop's baseline power consumption.

Using our load generating application, we first ran each benchmark subroutine with a single thread. We then continuously doubled the number of threads up to 32. We observed during the execution of our experiments that running 32 threads on all benchmarks subroutine reached peak processor utilization on both the phones and the desktop.

The total data gathered consists of three trials ran per thread-count per benchmark subroutine on each device. That is, the three devices ran each benchmark subroutine three times for each of 1, 2, 4, 8, 16, and 32 threads. To make the data more meaningful, we subtracted the constant baseline power drawn by the phones and desktop. From the Watts-Up meter data, we were able to calculate average power in Watts drawn during computation, and average total energy in Joules required to execute the benchmark.

A. Time Results

First, we looked at the time it took to complete each of the benchmarks. Figure 1 shows the time in minutes that it took each device to complete the matrix multiplication benchmark for 1, 2, 4, 8, 16 to 32 threads. As shown, the time remains relatively constant within a device up through 8 threads. Then, the computation time increases as the thread count increases. This growth is the result of threads competing for time on the CPUs. Each of the phones has eight cores so after 8 threads, the processes must wait for processor time. Though it is difficult to observe due to the scale of the chart, the desktop begins to experience this increase in time after 4 threads (which is the number of cores available). Actually, at 32 threads it took the desktop 8.62 times longer than its trials on 1 thread; whereas, the Moto x^4 took 5.49 times longer on 32 threads than 1 thread.

It is interesting to note that the Moto g^5 Plus, performs notably worse than the Moto x^4 for the matrix multiplication benchmark. It is consistently slower for all of the benchmarks. Most likely, this is due to the lower clock rate of its processors. However, for matrix multiplication, it averages 3.01 times

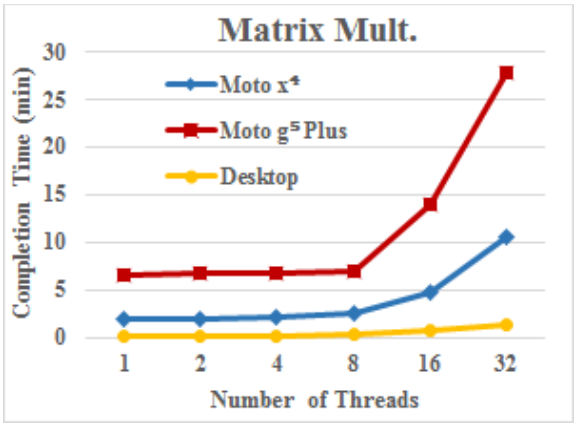

Fig. 1. The time it takes to complete the matrix multiplication benchmark for each device as a function of the number of threads.

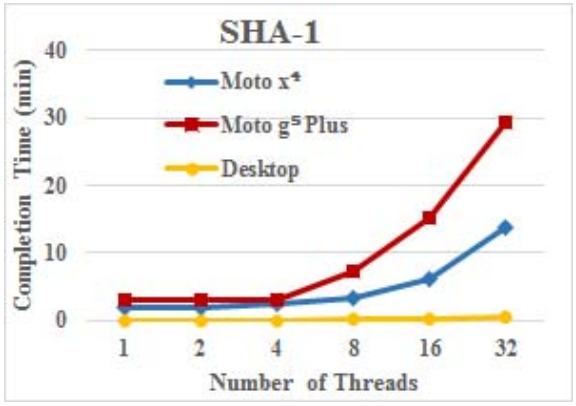

Fig. 2. The time it takes to complete the cryptographic hash benchmark for each device as a function of the number of threads.

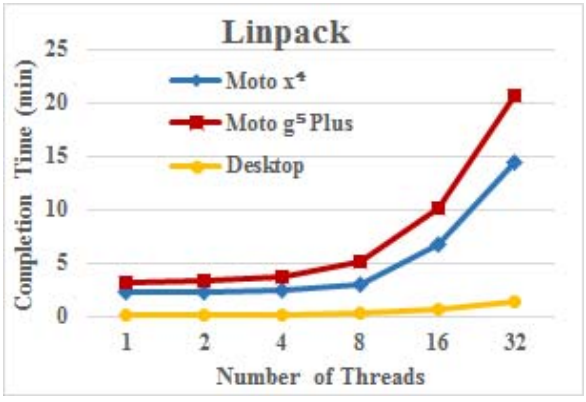

Fig. 3. The time it takes to complete the floating-point systems of equations benchmark for each device as a function of the number of threads.

slower than the x^4. For the SHA-1 benchmark, it averages 1.82 times slower, and for Linpack, it averages 1.51 times slower. One possible explanation for considerable slowdown might be that matrix multiplication is a memory-heavy task, and the Moto g^5 Plus has single channel RAM which is significantly slower than the dual channel RAM of the x^4.

For the SHA-1 and Linpack benchmarks (see Figures 2 and 3), the execution time for the x^4 and the g^5 phones are much closer until they are run on 8 threads. One possible explanation for this divergence is again related to the different memories. As threads begin to compete for resources, there may be less thrashing in the X^4, as it is able to satisfy memory requests more quickly.

It is unsurprising to observe that the desktop is always much faster than the phones. It completed the tasks on average 33.5 times faster than Moto g^5 Plus and 17.2 times faster than the Moto x^4. The desktop even performed the SHA-1 benchmark 50.6 times faster than the Moto g^5 Plus. The Moto x^4 stands a little more comparable on matrix multiplication by performing only 10.2 times slower than the desktop. The speed difference may be for a few reasons. The EliteDesk 800 G1 holds much more RAM with 16 GB than the phones with 3 and 4 GB. Its Intel cores clocking at 3.5 GHz is also significantly faster. In addition, there is a major difference in design of smartphone and desktop CPUs. The ARM System-on-Chip processors are optimized for small instruction sets (RISC), whereas Intel chips are designed to work x86, a CISC architecture. In general, desktops do not run on battery and have room for fans and heat sinks, which enables them to run at higher speeds and draw more power. There is currently no way for a smartphone to compete with a same generation desktop on time benchmarks.

B. Energy Results

A much more interesting comparison than timing is the total energy used to complete a task. Figure 4 shows the energy in Joules used to complete the matrix multiplication tasks per phone/desktop for each thread-count. It is important to remember that the multi-threaded applications are not collaborative. An additional thread is an entire discrete and identical benchmark running at the same time. As shown, the energy consumption of the Moto g^5 Plus is highest across all tasks (We ran comparisons on all threads, but only show 1, 4, 8, and 32 trials for space reasons).

It takes up to 3.3 times more energy than the desktop and up to 4.1 times more energy than the Moto x^4, which consistently took the least energy. A combination of similar power draws between the two phones and the Moto x^4's faster computation time explains why its energy consumption is lower than the Moto g^5 Plus. The Moto x^4 also uses lower frequency cores which require lower voltage. Since dynamic power consumption decreases with the square of the voltage, running a process on many low-frequency cores takes less overall energy than running the same process on fewer high-frequency cores. This is likely responsible for the energy savings of the Moto x^4 compared to the desktop which holds

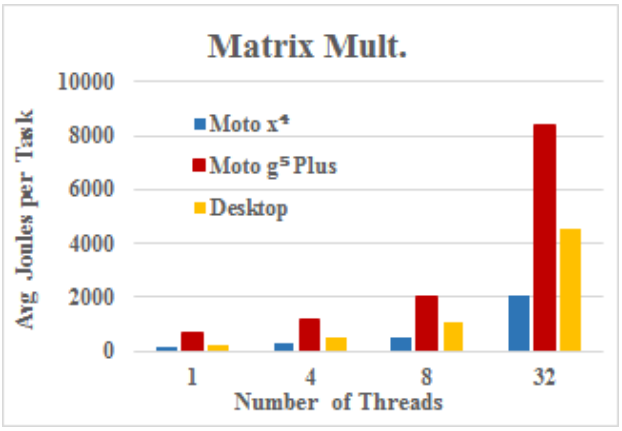

Fig. 4. The energy required to complete the matrix multiplication benchmark for each device with respect to the number of threads.

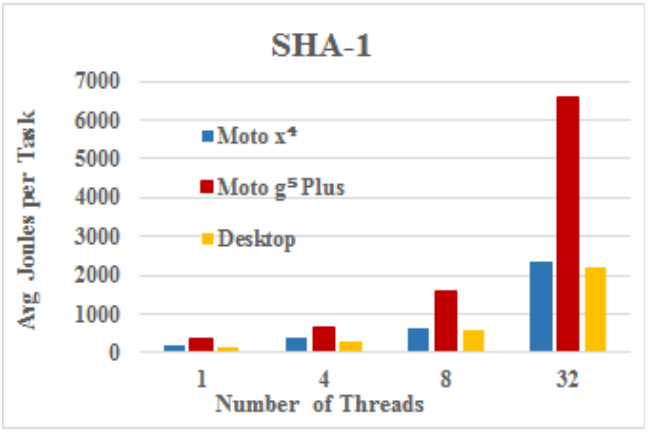

Fig. 5. The energy required to complete the cryptographic hash benchmark for each device with respect to number of threads.

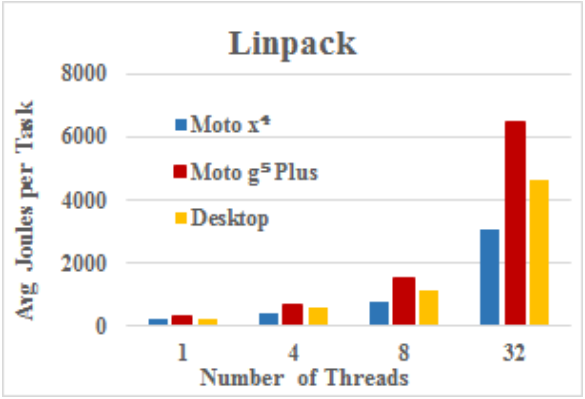

Fig. 6. The energy required to complete the floating-point systems of equations benchmark for each device with respect to the number of threads.

fewer and speedier processors. Running 32 threads of matrix multiplication on the Moto x^4 takes on average 54% less energy than on the desktop.

Figure 5 shows the same energy results but for the Secure Hashing Algorithm 1 benchmark. The Moto g^5 Plus again takes the most energy across all trials. However, the energy consumption of the Moto x^4 and the desktop are much closer for this task. The speed difference between the desktop and Moto g^5 Plus was greatest for the SHA-1 trials. The desktop performed the hashings at an average of 37% of the speed of the Moto g^5 Plus while the matrix multiplication took the desktop 47% of the Moto g^5 Plus's time. One possible reason for the desktop's fast hashing is that its CPU may be more tuned for this type of algorithm. The large decrease in the desktop's relative computation time causes its overall energy consumption to be quite low in this benchmark. In fact, the desktop actually uses the least amount of energy of the tested devices. On a single thread, it uses 35% less energy than the Moto x^4. However, as the threads increase, the energy savings between the desktop and Moto x^4 shrinks to just 6.5% at 32 threads.

The Linpack energy results, shown in Figure 6, are similar to the matrix multiplication energy results. The Moto g^5 Plus used the most energy, the Moto x^4 used the least, and the desktop's energy falls close to directly in-between. One interesting result of the Linpack trials is that the variability across the three devices is diminished. The desktop and Moto x^4 used about 73% and 54% of the energy that the Moto g^5 Plus used, respectively. Whereas in the matrix multiplication, the desktop and Moto x^4 used on average 46% and 25% of the energy consumed by the Moto g^5 Plus. The Linpack benchmark is the only trial that heavily uses floating-point operations. Since the two phones use the same type of processor cores, their potential floating point operations per second (FLOPS) are tied directly to the clock speed of their cores. This might be the reason for the phones' similar yet distinct performances. The desktop's high-frequency cores are again possibly responsible for its energy cost. Its fast performance makes it more energy efficient than the Moto g^5 Plus, but its high clock speeds might be the reason it requires more total energy than the Moto x^4.

C. Power Results

The average wattage drawn by the Moto x^4 during benchmark computation is graphed in Figure 7. A general trend can be observed. Power draw generally increases from 1 to 8 threads and decreases to 32 threads. We speculate that this may be due to the OS throttling the CPU to avoid overheating. The inconsistent relative heights of the bars indicate that more research is needed to explain the variability.

The desktop's power data is graphed in Figure 8. Its results are very consistent with what one would expect. The power increases as the number of threads increases and a core's become fully engaged. Starting at 4 threads, the power draw remains constant within benchmarks. The desktop only has 4 cores, and when they are all maxed out, additional threads wait and compete for CPU time. The desktop draws much more power than a smartphone, but it also processes much quicker.

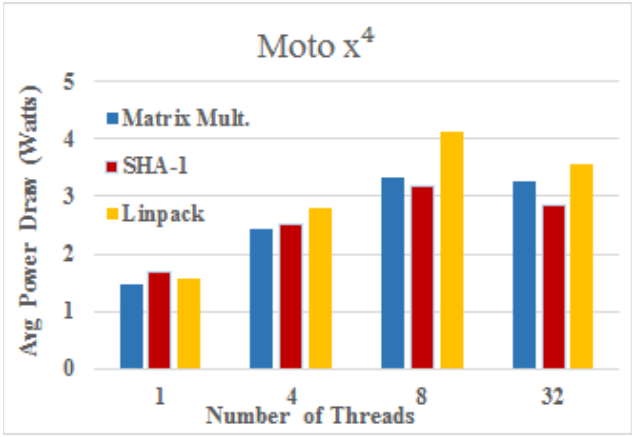

Fig. 7. The average power drawn by the Moto x^4 during each benchmark with respect to the number of threads.

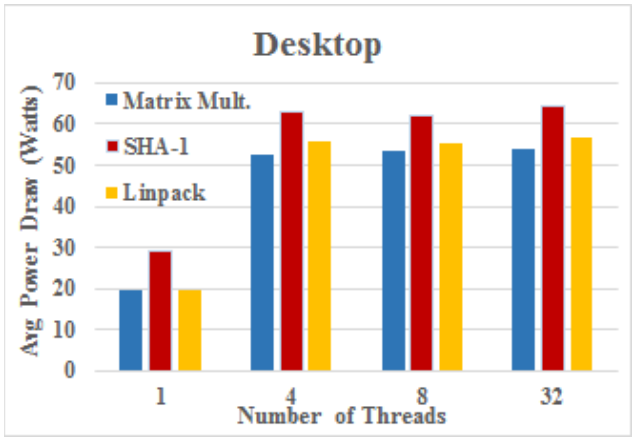

Fig. 8. The average power drawn by the desktop during each benchmark with respect to the number of threads.

D. Summary

Considering the energy and time results together yields interesting conclusions. The desktop always completed its tasks the fastest. The Moto g^5 Plus always took the longest time and most energy to complete its benchmarks. The Moto x^4 was always faster than the Moto g^5 Plus and slower than the desktop. However, the Moto x^4 also consumed the lowest amount of energy to perform identical tasks overall. On the other hand, the speedy desktop is arguably the least green simply due to its emergy. Even when the desktop is compared to the Moto g^5 Plus which draws more energy per task, the saved energy is not enough to make up for the difference in the energy cost of production. For example, the Moto g^5 Plus performed the most poorly running 32 threads of SHA-1. It used on average 6569 Joules while the desktop only used on average 2206 Joules. That is a difference of 4363 Joules. Now recall that the embodied energy of a mid-range smartphone

is about 1 GJ, which is 6.5 GJ less than the average energy to produce a desktop [11]. Dividing 6.5 GJ by 4363 J gives about 1.5 million. It would take 1.5 million benchmark runs on the Moto g^5 Plus to use up the surplus energy saved by not producing a desktop. Given that it takes the Moto g^5 Plus on average 29.4 minutes to run this benchmark, one would have to constantly run a Moto g^5 Plus for 83.3 years before producing a new desktop would be a justifiable alternative.

V. Conclusion and Future Works

In this paper, we presented an initial energy analysis of incorporating smartphones into the cloud as engines of computing. We created a load generating and power monitoring system that we deployed on two mid-range phones (Moto g^5 Plus and Moto x^4) and a mid-range desktop (Hewlett-Packard EliteDesk 800 G1). We conducted an empirical study that generated three different types of loads on our devices. Unsurprisingly, the desktop was able to complete the tasks averaging between 10 to 30$\times$ faster than either of the phones. However, for two of our benchmark loads, the Moto x^4 used up to 70% less energy than the desktop. For the third benchmark, the desktop used less energy than the Moto x^4, but under peak load, there was only a 6.5% difference.

Comparing the two phones also produced interesting results. The phones are reasonably similar in price and specifications, yet they produced very different results in our study. The Moto g^5 Plus used the most energy and time overall, while the Moto x^4 was on average the most energy efficient device. In the worst case, the Moto g^5 Plus used approximately 4$\times$ more energy to complete a task than the Moto x^4.

The results of this preliminary study indicate the need for additional research. In the future, we plan to further explore the impact of the slight difference between the two phones, including the energy trade-offs of LPDDR4 vs. LPDDR3 RAM. A natural extension of our work would be to increase the number and type of smartphones, desktops, and benchmark applications being compared.

References

[1] A. Binkley, "Customer focus on energy efficiency driving green data center market," October 2017. https://datacenterfrontier.com/customer-focus-on-energy-efficiency-driving-green-data-center-market/ [Online; posted 2-October-2017].

[2] A. Shehabi, S. J. Smith, D. A. Sartor, R. E. Brown, M. Herrlin, J. G. Koomey, E. R. Masanet, N. Horner, I. L. Azevedo, and W. Lintner, "United States Data Center Energy Usage Report," tech. rep., 06/2016 2016.

[3] A. L.A. Di Salvo, F. Agostinho, C. Almeida, and B. F. Giannetti, "Can cloud computing be labeled as "green"? insights under an environmental accounting perspective," vol. 69, pp. 514–526, 03 2017.

[4] D. P. Anderson, "BOINC: A System for Public-Resource Computing and Storage," in *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, GRID '04, pp. 4–10, 2004.

[5] P. Paul, "SETI @ Home Project and Its Website," *Crossroads*, vol. 8, pp. 3–5, Apr. 2002.

[6] U. of California, "Computing power," June 2018. https://boinc.berkeley.edu/ [Online; posted 6-June-2018].

[7] B. Walton, B. Perkins, and P. Lee, "The deloitte consumer review digital predictions 2018," Technical Report, Deloitte LLP, 2018.

[8] M. Y. Arslan, I. Singh, S. Singh, H. V. Madhyastha, K. Sundaresan, and S. V. Krishnamurthy, "Computing While Charging: Building a Distributed Computing Infrastructure Using Smartphones," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, pp. 193–204, 2012.

[9] R. Sanders, "New app puts idle smartphones to work for science," July 2013. [Online; posted 22-July-2013].

[10] "HTC power to give." http://www.htc.com/us/go/power-to-give/. Accessed: 2017-03-28.

[11] B. Raghavan and J. Ma, "The energy and emergy of the internet," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, pp. 9:1–9:6, 2011.

[12] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," in *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, pp. 21–34, 2010.

[13] W. L. Bircher and L. K. John, "Complete system power estimation using processor performance events," *IEEE Transactions on Computers*, vol. 61, no. 4, pp. 563–577, 2012.

[14] R. Murmuria, J. Medsger, A. Stavrou, and J. M. Voas, "Mobile application and device power usage measurements," in *2012 IEEE Sixth International Conference on Software Security and Reliability*, pp. 147–156, June 2012.

[15] T. Mudge, "Power: a first-class architectural design constraint," *Computer*, vol. 34, pp. 52–58, Apr 2001.

[16] LG Electronics Mobile Communications Company, "Lg launches worlds first and fastest dualcore smartphone," 2010.

[17] S. Harizopoulos and S. Papadimitriou, "A Case for Micro-cellstores: Energy-efficient Data Management on Recycled Smartphones," in *Proceedings of the Seventh International Workshop on Data Management on New Hardware*, pp. 50–55, 2011.

[18] S. Schildt, F. Busching, E. Jorns, and L. Wolf, "CANDIS: Heterogenous Mobile Cloud Framework and Energy Cost-Aware Scheduling," in *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, pp. 1986–1991, Aug 2013.

[19] F. Busching, S. Schildt, and L. Wolf, "DroidCluster: Towards Smartphone Cluster Computing – The Streets are Paved with Potential Computer Clusters," in *2012 32nd International Conference on Distributed Computing Systems Workshops*, pp. 114–117, June 2012.

[20] M. Y. Arslan, I. Singh, S. Singh, H. V. Madhyastha, K. Sundaresan, and S. V. Krishnamurthy, "CWC: A Distributed Computing Infrastructure Using Smartphones," *IEEE Transactions on Mobile Computing*, vol. 14, pp. 1587–1600, Aug 2015.

[21] T. U. Kumar and R. Senthilkumar, "CWC *– Secured distributed computing using Android devices," in *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*, pp. 1–7, April 2016.

[22] A. Paunikar, "LPDDR4: What Makes It Faster and Reduces Power Consumption," September 2017. https://blogs.synopsys.com/vip-central/2017/09/05/lpddr4-what-makes-it-faster-and-reduces-power-consumption/ [Online; posted 2-September-2017].

[23] A. Phillips, "ARM Launches Cortex-A50 Series, the Worlds Most Energy-Efficient 64-bit Processors," October 2012. https://www.arm.com/about/newsroom/arm-launches-cortex-a50-series-the-worlds-most-energy-efficient-64-bit-processors.php [Online; posted 30-October-2012].

[24] ARM, *ARM Cortex-A53 MPCore Processor Technical Reference Manual*. ARM.

[25] D. Eastlake, 3rd and P. Jones, "US Secure Hash Algorithm 1 (SHA1)," 2001.

[26] J. J. Dongarra, "Performance of various computers using standard linear equations software," *SIGARCH Computer Architecture News*, vol. 20, pp. 22–44, June 1992.