The Advantage of Computation Offloading in Multi-Access Edge Computing

Raghubir Singh†, Student Member, IEEE, Simon Armour†,
Aftab Khan‡, Mahesh Sooriyabandara‡, and George Oikonomou†
Communication Systems & Networks Research Group, University of Bristol, Bristol, UK †
Department of Electrical and Electronic Engineering, University of Bristol, UK†
Telecommunications Research Laboratory, Toshiba Research Europe Limited, Bristol, UK ‡
E-mail: raghubir.singh@bristol.ac.uk

***Abstract*—Computation offloading plays a critical role in reducing task completion time for mobile devices. The advantages of computation offloading to cloud resources in Mobile Cloud Computing have been widely considered. In this paper, we have investigated different scenarios for offloading to less distant Multi-Access Edge Computing (MEC) servers for multiple users with a range of mobile devices and computational tasks. We present detailed simulation data for how offloading can be beneficial in a MEC network with varying quantitative mobile user demand, heterogeneity in mobile device on-board and MEC processor speeds, computational task complexity, communication speeds, link access delays and mobile device user numbers. Unlike previous work where simulations considered only limited communication speeds for offloading, we have extended the range of link speeds and included two types of communication delay. We find that more computationally complex applications are offloaded preferentially (especially with the higher server:mobile device processor speed ratios) while low link speeds and any delays caused by network delays or excessive user numbers degrade any advantages in reduced task completion times offered by offloading. Additionally, significant savings in energy usage by mobile devices are guaranteed except at very low link speeds.**

Keywords*—Computation offloading, Multi-Access Edge Computing, task completion time, mobile devices, energy savings.

I. Introduction

A. Background

Although data processing capabilities for mobile devices such as smartphones have increased greatly, the energy stored in their batteries has failed to keep pace [1]. There has been considerable theoretical and experimental interest in using computation offloading to cloud resources (Mobile Cloud Computing, MCC) to leverage greater computing power while extending battery lifetime [2]–[8]. The traditional centre cloud such as (Amazon EC2 cloud, Microsoft Windows Azure, or Rackspace) can remain "unlimited" storage capacity and computing resources, reduced capital expenditure and minimized carbon footprints. However, this technology faces key issues: security, speed of services and slow connections, which are often combined as low link speed and high latency as mobile devices offload c omputational a nd p rocessing capacity to cloud computing services. These challenges have been exacerbated by the continued proliferation of mobile and fixed internet-connected devices. To overcome these disadvantages, Edge Computing, also known as "cloud at the edge," brings enhanced computing resources closer to the mobile user to minimize round trip times, which include transmission, queuing, processing and return times [9]. Furthermore, Edge Computing enables a faster dialogue between mobile devices and proximal Edge Computing servers and this is a particular advantage for time-critical applications [10]–[12]. MEC recognizes the importance of developing new applications for the users of mobile devices in a global marketplace where a heterogeneous populations of mobile devices with widely differing computing capacities, memory storage and battery lifetimes compete for access to computing resource [13]. The basic architecture of computation offloading with a MEC network is shown in Fig 1, where multiple users communicate with a base station and thence to a MEC server.

B. Motivation

The aim of this paper was to include the effects of multiple factors - on-board and MEC processor speeds, computational task complexity, a wide range of communication speeds, link access delay and the number of mobile users - on the success of offloading using task completion time as the sole criterion. In addition, the work aimed to quantify the reduction in energy use by mobile devices that were possible by offloading to MEC servers with different processor speed combination with varying communications link speeds.

C. Related Work

In this section, we briefly discuss frameworks devised to offload computations from smartphones and other mobile devices to cloud computing resources. In [5], the MAUI framework developed offloading strategies to improve mobile phone battery lifetimes with applications that included face recognition, video and chess games. The Cuckoo framework developed an algorithm for mobile devices running object recognition and real-time gaming applications [6]. Clonecloud [14] focused on improved performance for virus scanning and image searching while Chroma [15] used language translation and speech-to-text applications to demonstrate its ability to partition applications for remote execution. The Spectra framework investigated offloading to cloud resources for a speech recognizer, a document preparation system and a natural language translator [16]. COCA – Computation Offload to Clouds using Aspect-Oriented Programming - automatically offloaded part of the computation

978-1-7281-1796-6/19/$31.00 ©2019 IEEE

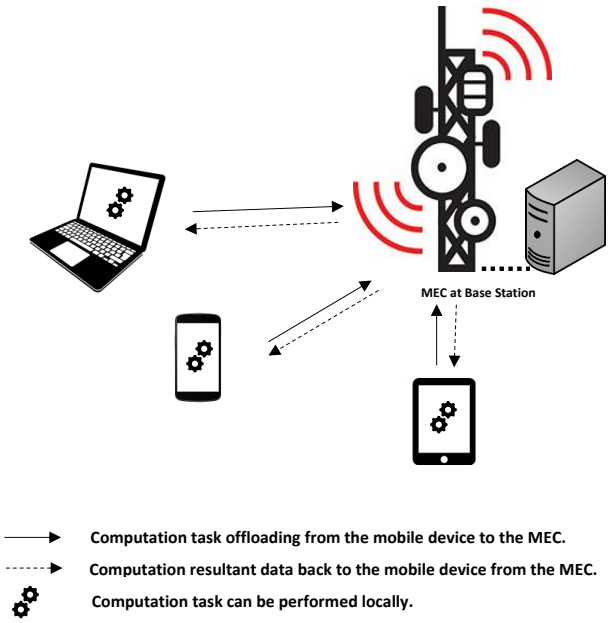

Fig. 1. System model of a MEC system for multiple users.

to cloud resources, resulting in improved performance and extended battery lifetimes [17].

Comparatively little research has been published concerning offloading in MEC networks but interest has increased recently. [18] focused on Internet of Things mobile devices while [19] considers how centralized cloud and MEC resources could be combined and [20] reported data from simulations of multiple MEC servers in 5G networks.

D. Contributions

The contributions made by this paper concern how a heterogeneous population of mobile devices with different MEC server-side processors in decisions affecting offloading, specifically:

- Effects of different mobile device on-board processor speeds on the achieving successful offloading for shorter task completion time.
- Effects of different MEC server processor speeds on the achieving successful offloading for shorter task completion time.
- Effects of widely differing data transmission speeds to the MEC servers on the achieving successful offloading for shorter task completion time.
- Modelling how link access delay and MEC server load caused by widely differing numbers of users reduces the success of offloading in achieving faster task completion time.
- Energy savings achieved by mobile devices with various combinations of on-board and server-side processor speeds with different data transmission speeds to MEC servers.

TABLE I. MAXIMUM BITS/INSTRUCTION FOR 9 APPLICATIONS

Applications	Bits/Instruction
siesta	5.29×10^{-5}
charmm	7.34×10^{-5}
mdrun_mpi	1.08×10^{-4}
nwchem	1.80×10^{-4}
vasp_ncl	2.86×10^{-4}
cocmomc	4.84×10^{-4}
lmp_stampede	9.53×10^{-4}
namd2	1.01×10^{-3}
fvcom	2.27×10^{-3}

All these factors are major parameters in the operation of functional MEC networks. The remaining sections of this paper present: the theoretical analysis and define the quantitative ranges of parameters included in the mathematical models of computational offloading for reduced task completion time and energy use by the mobile device (Section II), impacts on task completion time and energy use by the mobile device of these parameters (Section III) and conclusions for functioning MEC networks servicing multiple mobile device users simultaneously (Section IV).

II. THEORETICAL ANALYSIS AND QUANTITATIVE MODELS FOR OFFLOADING TO A MEC NETWORK

A. Offloading for Improved Execution Speed

In an MCC model [21] for offloading to result in a faster execution time for a task the following inequality was required:

$$\Gamma\left(\frac{1}{e}-\frac{1}{E}\right)>\frac{F}{C} \tag{1}$$

where Γ is the link speed (bps), e is the execution rate of the local computing device in instructions per second (IPS), E is the execution rate of the server in instructions per second (IPS), F is the data (bits) transferred over the MEC network and C is the size of the computational job (instructions) and the units for both sides of the equation are bits per instruction. If the left-hand side exceeded the right-hand side, computation offloading was favorable, i.e. the achieved task execution was faster by offloading to the external MEC server.

This inequality was a contraction of a more general inequality:

$$\Gamma\left(\frac{1}{e}-\frac{1}{E}-\frac{H}{C}\right)>\frac{F}{C} \tag{2}$$

where $\frac{H}{C}$ was the time per instruction that degrades the performance of the offloading system as a result of communication problems if $H > 0$. The authors of the study [21] equated H to zero and only considered an uncongested network but we have considered multi-user congestion in our analysis. The F/C term in equations 1 and 2 refers to bits per instruction values computed for the 9 applications listed in Table I, data from [21]. The F/C term is inversely proportional to the computational complexity of the application.

The processor speeds considered are: (for on-board devices) the Texas instrument's MSP430 and Apple A9 (1.6×10^7 IPS and 3.6×10^9 IPS, respectively) and (for server-side processors) either the Intel Celeron or Xeon processor (6.40×10^9 and 1.40×10^{11} IPS, respectively). [21] only considered two link speeds, 1 kbps and 1 Mbps for offloading. Here, we extend the range of link speeds up to 64 Mbps and include two types of communication delay: link access delays independent of the number of users and a user number-dependent reduction of the 1/E term in equation (1).

B. Offloading for Reduced Mobile Device Energy Usage

[22] presented an outline mathematical model for computing energy saving by offloading which relied on the inequality that the energy used by the mobile device was more than the energy of that mobile device in offloading; this can be written as

$$\frac{C \times P_c}{M} > \frac{C \times P_i}{S} + \frac{F \times P_{tr}}{\Gamma} \tag{3}$$

where the power terms for the mobile device are taken from [22], S represents the processing speed of the server and M the processing seed of the mobile device; other symbols have the definitions used for equations (1) and (2). F is considered to be the dominant contributor to any data exchange between the mobile device and MEC server, i.e. relatively little data is transmitted back to the mobile device but F may represent a large data file.

The authors of [22] quoted three values for power ratings (energy usage) by a mobile device: P_c is the energy consumption of the mobile device while computing (0.9 W), P_i is the energy consumption of the device while idling (0.3 W) and P_{tr} is the energy consumption (W) of the device while transmitting and receiving information (1.3 W). These values have been used in the calculations of the energy used by a mobile device computing locally or offloading to a MEC server. If the left-hand side of equation (3) exceeds the right-hand side, the energy use by the mobile device will be reduced by offloading to the MEC server. This will be advantageous to the user of the mobile device in, for example, extending the battery life.

III. Effects of Link Speed, Processor Speed and MEC Server Load on Offloading

A. Offloading for Improved Execution Speed

Fig. 2 shows the slower on-board processor (MSP430) with the slowest of the two server-side processors (Celeron, with a server-side:on-board processor speed ratio of 402:1). The calculated bits per instruction values were favourable for offloading for shorter task completion times for all 9 applications at a link speed of 64 kbps or more. With even a low link speed of 1 kbps, the more computationally complex application siesta was offloaded. In contrast, Fig. 3 shows that the faster (A9) on-board processor required much faster link speeds to justify offloading (in order to achieve lower task completion time): 33 Mbps with the slower (Celeron, with a server-side processor speed ratio of 1.7:1) and 16.4 Mbps for the faster (Xeon server-side processor speed ratio of 39:1). In general, the higher the bits per

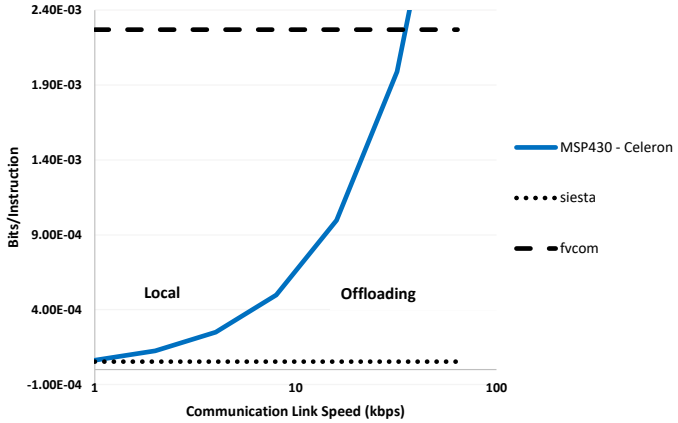

Fig. 2. Effect of link speed on the offloading threshold for shorter task completion time: MSP430 on-board processor offloading to the Celeron server-side processor; siesta and fvcom are the most computationally complex applications and the least, respectively.

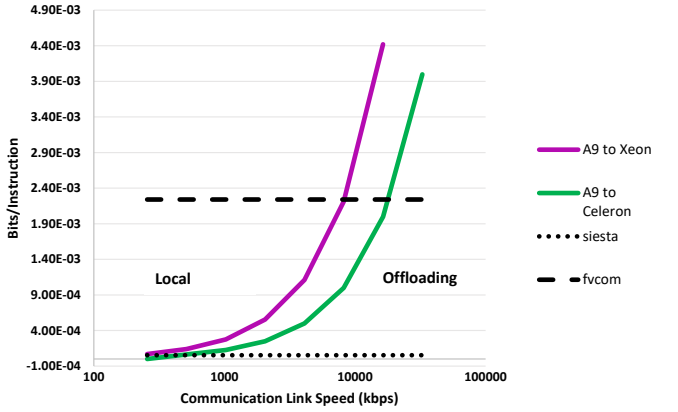

Fig. 3. Effect of link speed on the offloading threshold for shorter task completion time: A9 on-board processor offloading to the Celeron or Xeon server-side processor; siesta and fvcom are the most computationally complex applications and the least, respectively.

instruction value of an application, the higher was the minimum link speed required for a shorter task completion time to be possible by offloading. Table II enumerates the required link speed for all 9 applications in Table I with three combinations of mobile and server processors presented in Section II-A.

Mobile devices with low-speed processors would, therefore, find offloading advantageous for shorter task completion time even when accessing wireless personal area networks with limited ranges and low link speeds 250 kbps), i.e. those specified in IEEE 802.15.4. Devices with faster on-board processors would benefit by offloading computations to MEC networks with link speeds comparable to current 4G and Wi-Fi networks [23], [24].

B. Offloading from Mobile Device with Different Processor Speeds

A necessary corollary of the results presented with different combinations of processors with varying speeds is that, as the proportion of faster on-board processor mobile users in the user population increases, the success of offloading for faster task completion at a constant link speed decreases; this is because,

TABLE II. MINIMUM LINK SPEED FOR OFFLOADING APPLICATIONS WITH DIFFERENT PROCESSOR COMBINATIONS FOR SHORTER TASK COMPLETION TIME

Application	MSP430 to Celeron (kbps)	A9 to Celeron (kbps)	A9 to Xeon (kbps)
siesta	0.8	433.6	195.9
charmm	1.2	601.6	271.9
mdrun_mpi	1.7	885.2	400
nwchem	2.9	1475.4	666.7
vasp_ncl	4.6	2344.3	1059.3
cocmomc	7.8	3967.2	1792.6
lmp_stampede	15.3	7811.5	3529.6
namd2	16.2	8278.7	3740.7
fvcom	36.4	18606.6	8407.4

TABLE III. MAXIMUM LINK ACCESS DELAY FOR OFFLOADING APPLICATIONS WITH DIFFERENT PROCESSOR COMBINATIONS FOR SHORTER COMPLETION TIME AT 20 MBPS

Application	MSP430 to Celeron (ms)	A9 to Celeron (ms)	A9 to Xeon (ms)
siesta	62.3	0.12	0.27
charmm	62.3	0.12	0.27
mdrun_mpi	62.3	0.12	0.26
nwchem	62.3	0.11	0.26
vasp_ncl	62.3	0.11	0.26
cocmomc	63.3	0.10	0.25
lmp_stampede	62.3	0.07	0.22
namd2	62.2	0.07	0.22
fvcom	62.2	0.01	0.16

with higher on-board processor speeds, the left-hand term in equation (1) decreases.

If the user population in range of a MEC base station and server is composed of equal numbers of devices with on-board processor speeds covered by the range in Section II-A, approximately 2% found offloading for shorter task completion time is possible at a link speed of 250 kbps, 12% found offloading for shorter task completion time advantageous at a link speed of 1 Mbps whereas 64% benefited at a link speed of 5 Mbps, as shown in Fig. 4. This analysis assumes a uniform distribution of processor speeds in mobile devices in the user population attempting to offload the application with the greatest computation complexity (the lowest F/C value, see equation 1 and Table I) to a MEC server with the Xeon processor.

When applications with lower bits per instruction were considered, the percentage of mobile devices successfully offloading for shorter task completion time increased at a fixed link speed. If the link speed exceeded 8.4 Mbps (Table II), all the mobile devices offloaded all the applications in (Table I) for a shorter task completion time. Mobile devices with faster on-board processors than those considered would require faster link speeds when offloading to the Xeon server-side processor.

C. Offloading with Link Access Delays

A positive H/C term in equation (2) adds a link access delay to the communication link between the mobile device and the MEC server and reduces the left-hand side of the equation until eventually the inequality shown in equation (2) fails and offloading does not result in short task completion times. Table III presents maximum computed link access delays for the 9 applications with three combinations of on-board and MEC server-side processors. With the MSP430 on-board processor in combination with any of the three server-side

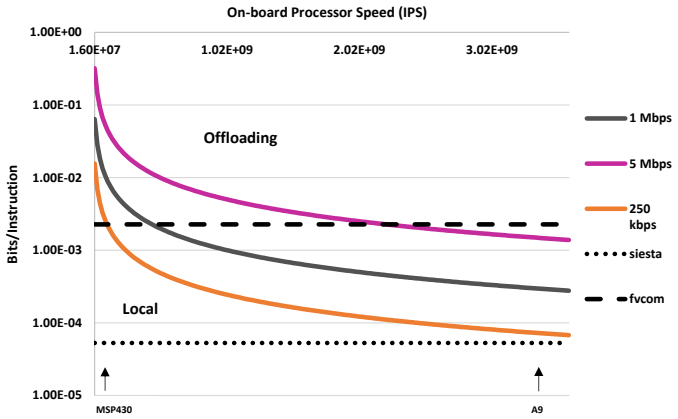

Fig. 4. Effect of on-board processor speed on computed bits per instruction values required for offloading to a MEC server-side processor (Xeon) at different link speeds: e.g. siesta app is always offloaded.

TABLE IV. MAXIMUM NUMBER OF MOBILE DEVICES OF OFFLOADING FOR SHORTEN COMPLETION TIME WITH DIFFERENT PROCESSOR COMBINATIONS

Application	MSP430 to Celeron (250 kbps)	A9 to Xeon (20 Mbps)
siesta	400	36
charmm	399	36
mdrun_mpi	398	36
nwchem	396	36
vasp_ncl	394	35
cocmomc	388	34
lmp_stampede	376	30
namd2	375	30
fvcom	342	21

processors, offloading required progressively higher bandwidths until, at a link access delay factor exceeding 62.3 ms per 10^6 instructions at a link speed of 20 Mbps, offloading failed entirely to result in a short task completion time at any link speed, as shown in Table III. The combination of the faster (A9) on board processor with the Celeron server-side processor did not tolerate link access delays factors greater than 0.12 ms per for 10^6 instructions, as shown in Table III, while the A9/Xeon combination failed to offload at any link speed when link access delays factor exceed 0.27 ms per 10^6 instructions. Faster on-board processors therefore required a less interrupted and more seamless communication link in order to make offloading beneficial for shorter task completion times.

D. Offloading with MEC server load

Ideally, any mobile device would have unimpeded access to the MEC server for offloading. When large numbers of users attempt to access the same MEC server, to avoid network overload, queuing and scheduling strategies have been proposed [25]. In the extreme case, an overloaded MEC server might also be able to share computational jobs with other servers [26].

To include an analysis of the effects of the number of mobile devices attempting to connect simultaneously to a MEC server, equation (2) was modified as:

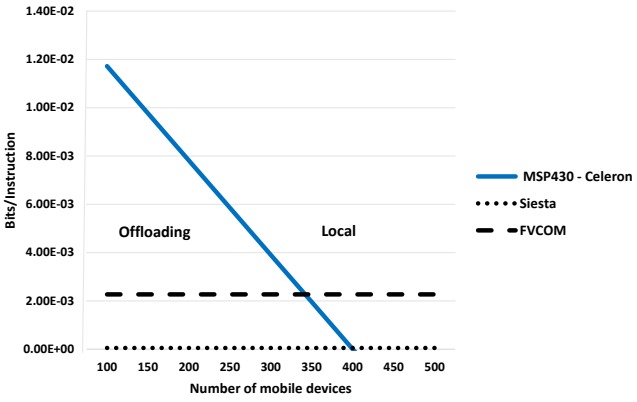

Fig. 5. Effect of number of mobile users on the offloading for shorter task completion time: MSP430 on-board processor offloading to the Celeron server-side processor;siesta and fvcom are the most and least computationally complex applications, respectively (Table I).

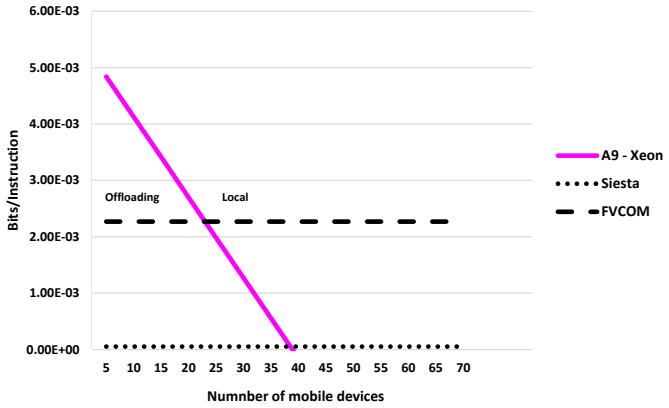

Fig. 6. Effect of number of mobile users on the offloading for shorter task completion time: A9 on-board processor offloading to the Xeon server-side processor; siesta and fvcom are the most and least computationally complex applications, respectively (Table I).

$$\Gamma\left(\frac{1}{e}-\frac{D}{E}\right)>\frac{F}{C} \tag{4}$$

where D represents the number of users; this introduces a reduction in communication link speed depend on the number of users. In effect, overloading the MEC servers reduced the ratio of the server:mobile processor speeds.For the slower MSP430 processor with the slower Celeron server processor, a link speed of 250 kbps was sufficient to offloaded most of the applications for up to 400 users, as shown in Table IV. With the faster A9 processor with the faster Xeon server processor, even a link speed of 20 Mbps only offloaded much smaller numbers of mobile users.

Fig. 5 shows that the most computationally complex application siesta was always offloaded for faster completion time from an on-board MSP430 processor to a MEC server (Celeron) until the number of mobile users exceeded 400 while the least computationally complex application (fvcom) was preferentially computed locally when the user number exceeded 342. Fig. 6 shows that the most computationally complex application siesta was always offloaded for faster completion time from an on-board A9 processor to a MEC server (Xeon) until the number of mobile users exceeds 36 while the least computationally complex application (fvcom) was preferentially computed locally when the user number exceeded 21. In general, the less computationally applications tolerated smaller maximum user numbers because they required higher differentials in the relative speeds of the server and on-board processors to achieve shorter task completion times (Table IV).

E. Offloading for Energy Saving

Using equation (3), combining the slower (MSP430) on-board processor with any of the three server-side processors resulted in major energy savings for the mobile device (up to 99%) at low link speeds (100-200 kbps) but the faster (A9) on-board processor required much faster link speeds for maximum energy savings (Fig. 7). Even with a relatively low link speed of 1 Mbps, an energy saving of 80% was possible with the A9/Xeon combination. Calculations showed that combining the

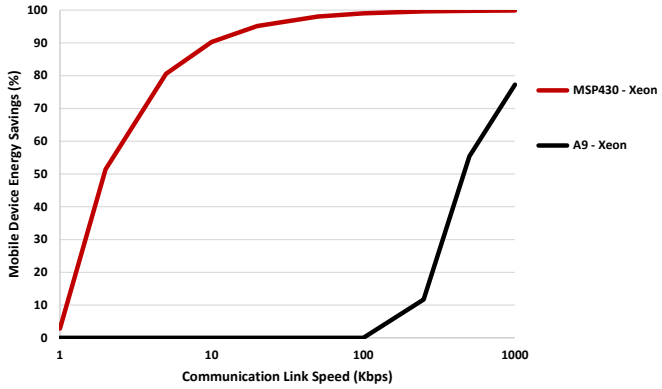

Fig. 7. Effect of link speed on energy savings by mobile devices with slow and fast on-board processors offloading to the MEC server (Xeon).

A9 processor with either of the two faster server-side processors could give energy savings for the mobile user exceeding 90% at high link speeds (50-100 Mbps).

IV. Conclusions and Future Research

The success of offloading in a MEC network for faster task execution is highly dependent on following four parameters: mobile device processor speed, MEC server processor speed, complexity of the task considered for offloading, and link speed. The combination of on-board and server-side processors is crucial in determining the minimum link speed required for offloading to result in shorter task completion times. Relatively slow on-board processors such as the MSP430 can offload to MEC servers at very low link speeds provide by low-bandwidth networks because the ratio of server-side processor speed to on-board processor speed is high. However, a MEC network should have the flexibility to adapt to widely fluctuating numbers of mobile devices seeking connections with different processor speeds and differing task complexities in an environment where available bandwidth could decrease rapidly. In addition, MEC service suppliers will increasingly be required to provide the greatest processor speed edge over currently available mobile devices together with the highest economically practical bandwidth for data transmission and return to ensure maximal user Quality of Experience (QoE).

The less complex the computational task, the greater is the minimum link speed required but this is also affected by the combination of device and server processors. Consequently, mobile device heterogeneity becomes an important factor, especially at times of low available bandwidth. Additionally, however, interruptions and delays in the ability of mobile devices to access MEC servers ("link access delays") greatly reduce the ability of offloading to offer benefits for task completion times. This problem is magnified if a higher-speed on-board processor is used in the mobile device. The load on a MEC network caused by increasing numbers of mobile users can erode any task completion time advantage by offloading as the differential between the MEC server and on-board processor speeds decreases. Nevertheless, the energy savings mobile devices by offloading to a MEC server are major even at low or modest available bandwidths in a 4G network. In practice, with fast server-side processors and high bandwidths, the default option for the users of mobile devices could be to offload primarily or solely for energy (battery lifetime) savings even if execution times were not improved by offloading because of the very marked effect on energy use by the mobile device. This not only increases the demand pressure to offload to the MEC system on the supplier side but also necessitates (on the user side) a decision-making process in which both task execution time and energy saving factors can be assessed.

A. Future Research

Future work will focus on developing a mathematical optimization model to minimize total completion time for tasks when factors such as CPU workloads are taken into consideration, extending the analysis to where offloading to more than one available MEC server is possible and factoring in any user price cost of using a MEC service.

V. Acknowledgement

This work was supported by the Engineering and Physical Sciences Research Council grant number EP/P510427/1, Toshiba Research Europe Limited, and the University of Bristol.

References

[1] V. Bahl. (2015) emergence of micro datacenter (cloudlets/edges) for mobile computing. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/Micro-Data-Centers-mDCs-for-Mobile-Computing-1.pdf

[2] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[3] C. Xian, Y.-H. Lu, and Z. Li, "Adaptive computation offloading for energy conservation on battery-powered systems," in *Proceedings of the 13th International Conference on Parallel and Distributed Systems*, vol. 1, pp. 1–8, 2007.

[4] R. Wolski, S. Gurun, C. Krintz, and D. Nurmi, "Using bandwidth data to make computation offloading decisions," in *2008 IEEE International Symposium on Parallel and Distributed Processing*, pp. 1–8.

[5] E. Cuervo, A. Balasubramanian, D.-K. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[6] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, "Cuckoo: a computation offloading framework for smartphones," in *Gris M., Yang G. (eds) Mobile Computing, Applications, and Services. MobiCASE 2010. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, vol. 76. Springer, Berlin, Heidelberg, pp. 59–79.

[7] A. R. Khan, M. Othman, A. N. Khan, J. Shuja, and S. Mustafa, "Computation offloading cost estimation in mobile cloud application models," *Wireless Personal Communications*, vol. 97, no. 3, pp. 4897–4920, 2017.

[8] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, 2015.

[9] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[10] S. Clinch, J. Harkes, A. Friday, N. Davies, and M. Satyanarayanan, "How close is close enough? understanding the role of cloudlets in supporting display appropriation by mobile users," in *2012 IEEE International Conference on Pervasive Computing and Communications*, pp. 122–127.

[11] K. Ha, P. Pillai, G. Lewis, S. Simanta, S. Clinch, N. Davies, and M. Satyanarayanan, "The impact of mobile multimedia applications on data center consolidation," in *2013 IEEE International Conference on Cloud Engineering (IC2E)*, pp. 166–176.

[12] W. Hu, Y. Gao, K. Ha, J. Wang, B. Amos, Z. Chen, P. Pillai, and M. Satyanarayanan, "Quantifying the impact of edge computing on mobile applications," in *Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems*. ACM, 2016, p. 5.

[13] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[14] 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.

[15] R. K. Balan, D. Gergle, M. Satyanarayanan, and J. Herbsleb, "Simplifying cyber foraging for mobile devices," in *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, 2007, pp. 272–285.

[16] J. Flinn, S. Park, and M. Satyanarayanan, "Balancing performance, energy, and quality in pervasive computing," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*. IEEE, 2002, pp. 217–226.

[17] H.-Y. Chen, Y.-H. Lin, and C.-M. Cheng, "Coca: Computation offload to clouds using aop," *12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp. 466–473, 2012.

[18] H. Guo, J. Liu, and J. Zhang, "Computation offloading for multi-access mobile edge computing in ultra-dense networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 14–19, 2018.

[19] H. Guo and J. Liu, "Collaborative computation offloading for multi-access edge computing over fiber-wireless networks," *IEEE Trans. Veh. Technol.*, vol. 67, pp. 4514–4526, 2018.

[20] H. Guo, J. Liu, and J. Zhang, "Efficient computation offloading for multi-access edge computing in 5g hetnets," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[21] S. Melendez and M. P. McGarry, "Computation offloading decisions for reducing completion time," in *14th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2017, pp. 160–164.

[22] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[23] J.-S. Lee, Y.-W. Su, and C.-C. Shen, "A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi," in *33rd Annual Conference of the Industrial Electronics Society (IECON)*, 2017, pp. 46–51.

[24] ETSI. (2018) Long term evolution. [Online]. Available: https://www.etsi.org/technologies/mobile/4g.

[25] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj, "Optimal schedule of mobile edge computing for internet of things using partial information," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2606–2615, 2017.

[26] D. Satria, D. Park, and M. Jo, "Recovery for overloaded mobile edge computing," *Future Generation Computer Systems*, vol. 70, pp. 138–147, 2017.