ACCEPTED FROM OPEN CALL

D2D Task Offloading: A Dataset-Based Q&A

Dimitris Chatzopoulos, Carlos Bermejo, Ehsan ul Haq, Yong Li, and Pan Hui

The authors present the main types of D2D applications, and discuss the characteristics of task offloading and the quality of experience in D2D ecosystems. They list five popular questions regarding the ability of other users to be helpful in terms of resources, connectivity, availability, incentives, and security.

Abstract

D2D interactions are promoted both in the context of traffic and computation offloading. Applications with functionalities that are executed in parallel in more than one mobile device have been proposed while the idea of using another mobile device as a relay is going to be part of 5G. In this work, we first present the main types of D2D applications, then we discuss the characteristics of task offloading and the quality of experience in D2D ecosystems. After that, we list five popular questions regarding the ability of other users to be helpful in terms of resources, connectivity, availability, incentives, and security. To answer the listed questions, we use two mobile big data datasets, one experimental study, and arguments from our experience and the literature.

Introduction

Nowadays, we use smartphones to do most of the tasks that we used to do on traditional computers. Although only 20 percent of the world's total population owned a smartphone in 2013, this figure was expected to increase to 34 percent by 2017 while in the U.S. four out of five mobile users are expected to use a smartphone. Moreover, their capabilities do not stop increasing, with more memory, more powerful CPUs, and more sensors every year. It is worth mentioning that the capabilities of an average modern smartphone exceed the capabilities of a conventional server of the previous ten years. However, these devices are only used for personal use by their owners. At the same time, developers continue building even more resource-hungry and data-hungry applications, in terms of computational requirements, memory, storage, Internet access, sensor requirements and energy needs.

Apart from the increasing requirements of computational resources, mobile users are continuously increasing their need for cellular bandwidth. This is due to the use of data-hungry mobile applications that download content from multiple content providers and the introduction of affordable data plans by cellular operators. According to the Global Mobile DataTraffic Forecast Update of Cisco for this year, mobile data traffic has grown 18-fold over the past five years, almost half a billion mobile devices and connections were added in 2016, and there exist 325 million wearable devices [1].

Mobile network operators are struggling to handle the increasing amounts of data traffic in their networks, and one approach to deal with this issue is *Mobile Data Offloading*. Mobile Data Offloading techniques in LTE networks use the available WiFi/WiFi Direct technologies in the unlicensed band to provide substantial aid by under-loading the operators' networks and improving the coverage. In more detail, instead of requiring every mobile user to connect to a cellular tower and download the required content, Mobile Data Offloading enables content downloading through relays. The relays can either be standard WiFi access points or other smartphones. In the former case, the offloading techniques are based on WiFi while in the latter on WiFi-direct. To cope with the requirements for more computational resources, *Computation Offloading* is seen as a solution for overcoming the need. This model of computation is called Mobile Cloud Computing (MCC). Many offloading frameworks run virtual machine (VM) surrogates of mobile devices in the cloud and use them for computation offloading [2, 3]. Mobile cloud computing (MCC) approaches offload the most computationally expensive parts of mobile applications to cloud surrogates to provide a better quality of experience to the end users. Advances in MCC have been mainly focused on the offloading decisions, the connectivity issues with the cloud surrogate, and pricing models. More recently, inspired by the success of message forwarding in Delay Tolerant Networks (DTN), researchers have proposed computation offloading solutions that rely solely on nearby mobile devices, a technique known as device-to-device offloading (D2D offloading) [4, 5]. The idea is simple: most of the time we are surrounded by mobile devices that can serve as thin remote servers and execute offloadable tasks of nearby mobile clients. The lower communication delay, lower packet loss, and higher bandwidth between the mobile and the proximal device make this solution more suitable than the public cloud in some circumstances. Existing research works show that not only is D2D offloading possible, but it is also beneficial regarding energy consumption and execution time for tasks that can tolerate small delays. Also, mobile applications that utilize nearby mobile devices to improve users' quality of experience (QoE) have been implemented. Some popular examples of such applications are cooperative streaming, context-aware applications, video compression and face recognition, peer-to-peer location-based privacy, and sensing. Figure 1 shows the complete ecosystem where mobile devices are connected to the cloud via either WiFi access points or via cellular access points, they can be connected to a satellite for location information, and they can be connected with each other.

Digital Object Identifier:
10.1109/MCOM.2018.1700873

Dimitris Chatzopoulos, Carlos Bermejo, Ehsan ul Haq, and Pan Hui are with Hong Kong University of Science and Technology; Pan Hui is also with the University of Helsinki; Yong Li is with Tsinghua University.

Task Offloading

Applications on D2D ecosystems can be of many types. Traditional packet forwarding and routing in DTNs will regain popularity on the arrival of 5G technologies because they allow users' traffic to be routed via other proximal mobile devices. Moreover, new smartphones will be equipped with more than one cellular transceiver and will be able to connect with multiple networks at the same time via the LTE-direct technology [6]. Furthermore, WiFi-direct and Bluetooth allow mobile devices to be connected in parallel with cloud infrastructures and with each other.

Also, applications can be executed in more than one device following the paradigm of computation offloading, which was initially proposed for MCC architectures. When a device receives a task from another node, it needs to allocate additional resources to process the task. For example, mobile crowd sensing applications make use of devices' sensors to perform local measurements and share their data with each other. The computational part of these applications is not massive, but it requires interconnectivity between the devices. Context-aware applications require help from other nearby devices to estimate the context. Context is a multifunctional variable of time and the ambient conditions; it is a type of information that is worth sharing between mobile devices regardless of whether they have past interactions or not. Resource sharing costs the device concerning battery and, in case of data plan sharing, regarding money. These costs can be expressed as a function of the needed resources and the network overhead. The requirements of the offloadable tasks can vary based on the functionalities of the application. The main difference between applications that have been introduced for mobile ad-hoc networks and those mentioned above is the variety in the possible requested help. We present a few types of applications in Table 1.

Quality of Experience in D2D Ecosystems

Each application has its own performance metrics, which depend on the local and the remote resources they are using for its execution as well as on the connectivity to the remote resources. Depending on the type of application, each resource has different significance. For example, applications with connectivity needs, like packet forwarding, evaluate the help of other users based on whether they forward the packets they receive toward the destination. On the other hand, for compute-intensive applications, like video compression [7], the help of a nearby device cannot be measured with the same simplicity for two main reasons: resource utilization and connectivity.

Resource Utilization: The resources that the helping device has to allocate for the execution of such offloaded task can be a substantial part of the total resources. It is possible for a computationally demanding task to require some minutes to be executed and this execution can cause a remarkable battery drop. For such cases, each mobile device has to consider its current state and the requirements of the offloaded task, and ideally estimate the execution time of the task and the battery drop.

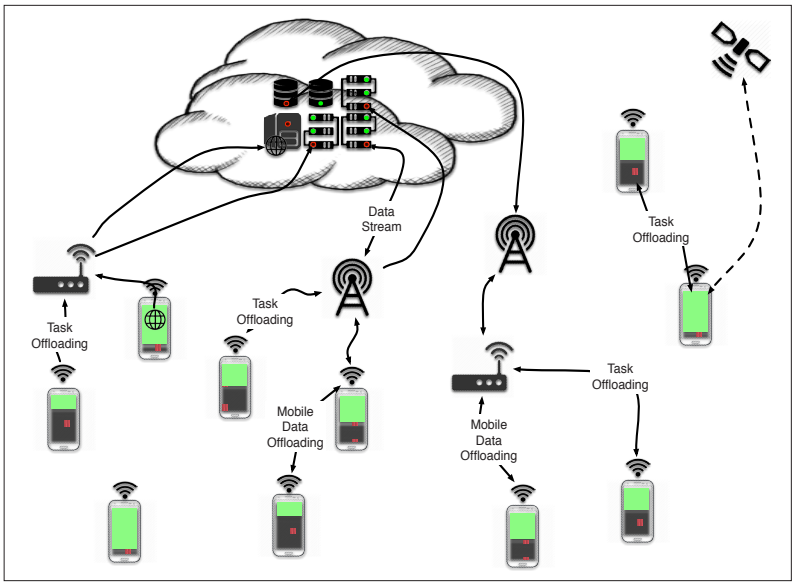

Figure 1. Mobile devices can be connected to each other, connect to WiFi access points and to cellular towers. Cellular providers can offload traffic through WiFi access points in order to release some of their resources. Mobile devices may use several types of services that are cloud based or in general offered by a remote server.

Application	Basic functionalities
P2P based k-anonymity location privacy service	Used by privacy-sensitive mobile users to protect their digital footprints. Users can exchange identities in location based services in order to preserve their anonymity.
Cooperative streaming	Improves users' QoE since it provides better video quality and combinations of individual libraries. Users cooperatively download videos or use their devices to create joint playlists.
Face recognition	Offers performance speedup and avoids the use of mobile cloud computing services that may also impose monetary costs.
Video compression	Offers performance speedup and avoids the use of mobile cloud computing services that may also impose monetary costs.
Sensing	Saves battery and increases accuracy.
Computation offloading	Saves energy and speeds up performance.
Context sharing	Mobile devices form groups and share contextual information.
Opportunistic grouping	Used to conserve power and share resources.
Opportunistic networking	Message forwarding.

Table 1. Examples of applications that are designed for D2D ecosystems.

Connectivity: Even if one device is willing to allocate part of its resources for the execution of a task that belongs to another device, the two devices should be able to be in contact for the transmission of the task and the execution result. In cases where delays in the return of the results can be tolerated, the two devices can disconnect during the execution phase and wait for another opportunity for connectivity in the near future.

Mobile Big Data

Mobile big data are described by the 3Vs (volume, variety and velocity) of big data literature and have attracted the interest of many parties from different perspectives [8] and are expected to attract even more interest due to the upcoming 5G cellular communication protocol [9].

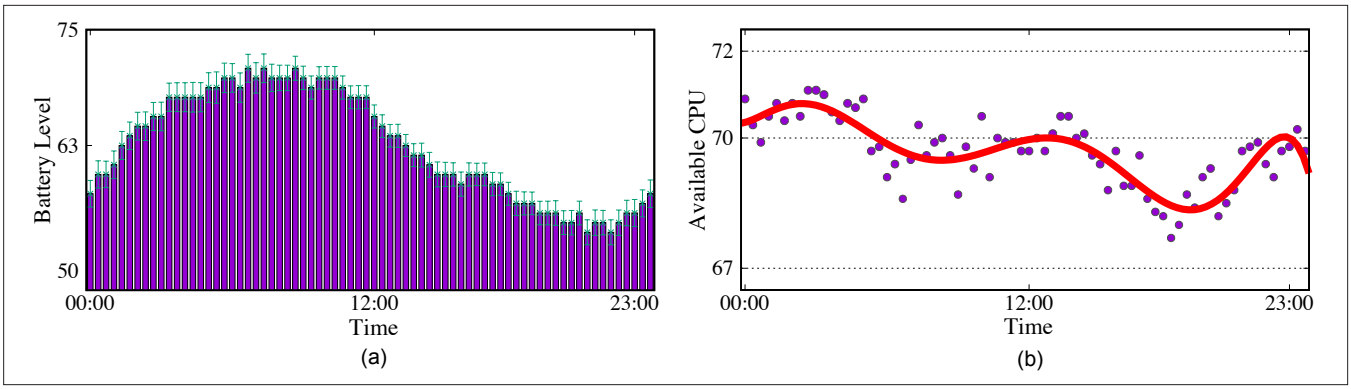

Figure 2. Average battery level (a) and available CPU cycles (b) of 263 devices within a day.

In this work, we employ two mobile big data datasets and one experimental study with 733 participants in order to answer the questions in the following sections. The two datasets are:

•Device analyzer dataset from the University of Cambridge that contains data regarding the usage of more than 30000 mobile devices [10]. In more detail, in this work we considered the collected data of 263 active mobile users. The data we used is the battery level of the devices, the time duration and connectivity patterns of the Bluetooth and WiFi interfaces as well as the CPU cycles.

•A mobile big data dataset that we accessed through a major telecom provider in China. The dataset contains web content requests from mobile users through the cellular interface of their mobile devices. The dataset includes 64,457,317 requests from 16,720 mobile users during the two-week period 1/3/2016 to 14/3/2016 from 2651 locations in the city of Shanghai. The examined data are anonymized and we cannot align the ID of the cellular towers to geographic locations in order to provide physical distance details and extract the users' mobility pattern. Each entry contains an anonymized identification of the mobile user, an anonymized identification of the cellular tower to which the mobile user was associated, a timestamp and the hostname of the request. The mobile users browsing the web, while using any means of transportation in the city of Shanghai, are associated with the cellular towers and the accessed website through their mobile devices.

The experimental study focuses on the effect of social ties in crowdsensing applications and has two parts. The first part uses the Facebook API to infer the social relationships between the participants; the second part is based on Amazon Mechanical Turk [11]. We are using these two datasets, the experimental study and related work to answer the following five questions:

- Can a neighbor be helpful?
- How can a neighbor be helpful?
- When is a neighbor able to help?
- Why would a neighbor be willing to help?
- Is it safe to execute my neighbor's task?

Question 1: Can a Neighbor Be Helpful?

In order for a mobile device to be helpful to another device, they should not harm the quality of experience of the mobile user that is executing the application and they have to allow the device that asks for help to consume fewer resources in comparison to the ones they would have used if the application was executed locally. Given that the ecosystem has many stochastic parameters, the condition above can be relaxed into the one that considers that on average a mobile device is helpful. The most common source of stochasticity is the interconnectivity between the devices. It is common for two devices to disconnect due to the mobility of their carriers, energy-saving policies, and interference from other devices.

However, given that a helping device should not be able to violate the quality of experience of its applications, they should have available resources to execute all of them smoothly. The two most common factors that affect the quality of experience are the battery level and the CPU level. Figure 2a shows the average battery level of 263 devices in the Device Analyzer dataset, and Fig. 2b shows the average available CPU fraction of the same devices during the day period. Time slots are divided into 20 minute intervals spanning 24 hours. Values against each time slot are an average value for all users that have valid data records within that time interval. Error bars show the standard error for each time instance. Both plots start at midnight. For each user, the free CPU resource values are calculated according to the number of cores on the particular device. As expected, most mobile users charge their devices once a day and usually while they are sleeping or when they return to their houses. The available CPU fraction peaks during sleep times and work times and it has local minima when the mobile users are commuting to and from their work. Both plots show that the average resource availability is at least 50 percent at any time.

Answer: Yes. Since at any time most mobile users have more than 55 percent of available battery and they are utilizing less than one-third of their CPU capabilities, we argue that resource-wise a neighbor can be helpful.

Question 2: How Can a Neighbor Be Helpful?

Given that a nearby device has the available resources, as we discussed in the previous section, the question we are trying to answer in this section is whether these available resources can be utilized. Technically, there are multiple ways for two devices to interconnect either in a single

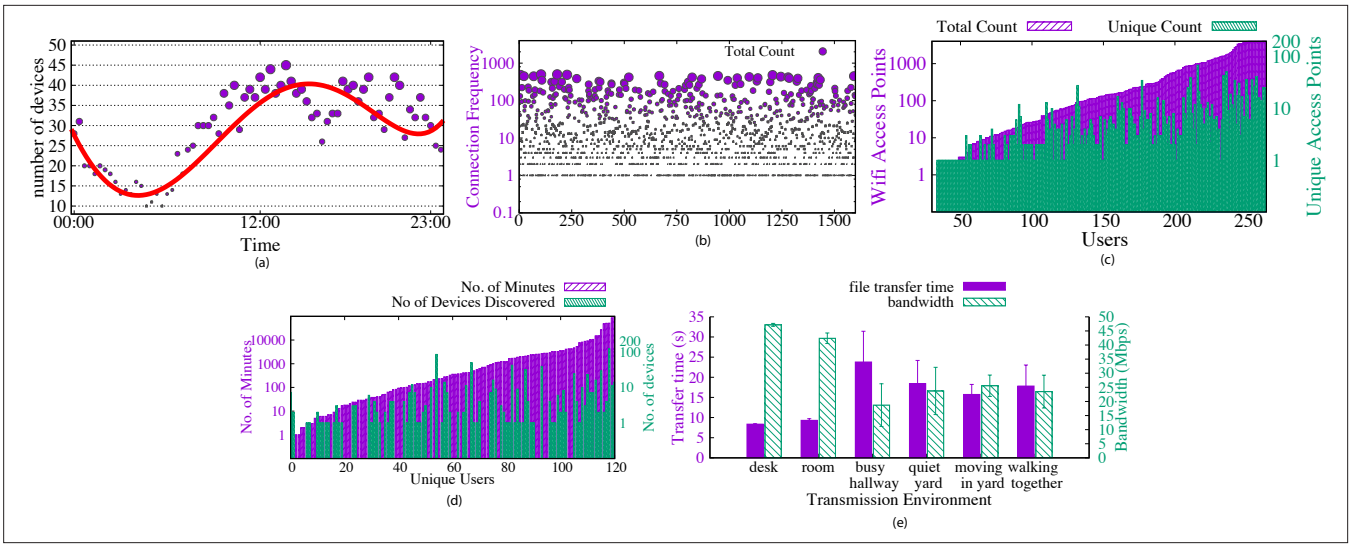

Figure 3. Available ways for two devices to connect with each other: a) Average number of activated Bluetooth interfaces over a day; b) Connection frequency of considered WiFi APs; c) Unique WiFi routers vs the total number of WiFi connections made for each user; d) Number of discovered devices and the time during which the Bluetooth interface was ON for 120 users; e) Transfer time of a 50 MB file and bandwidth of WiFi-direct for 6 different transmission environments.

hop (WiFi-direct, Bluetooth, NFC, and so on) or using a WiFi or a cellular access point.

Figure 3a shows the number of devices that have turned Bluetooth on over the period of 24 hours. Each entry is recorded for 20 minute periods, resulting in three buckets for each hour. As expected, during sleep hours, Bluetooth is turned ON in a small number of devices, but in the afternoon this number is four times higher. Figure 3b shows the connection frequency of 1600 WiFi access points. On the y-axis, it shows the number of times, in log scale, a device connected on a particular WiFi router. Figure 3c compares the unique WiFi routers and the total number of WiFi connections made for each user. The left y-axis shows the total number of times a user has connected to WiFi access points. The same WiFi access point may appear multiple times. The right y-axis shows the number of unique WiFi access points. Similarly, Fig. 3d shows the comparison between the number of discovered devices during the time Bluetooth was on for some particular user. The total number of users that have their Bluetooth activated (ON) at any time is 120. The left y-axis on the chart shows the duration in minutes for which the Bluetooth was on while the right y-axis shows the number of devices discovered. Figure 3e shows a set of experiments with two Xiaomi Mi3 devices that were used to transfer a file of 50 MB between the two devices to measure the bandwidth and the transmission time of the file. The devices were interconnected via WiFi-direct. Each experiment was repeated 50 times, and we considered six transmission environments. The average and the standard deviation are presented. In static transmission environments where the devices are not moving and are placed on the same desk or in the same room, and there are only a few other mobile devices that can create interference, the standard deviation is minimal, while in cases where the devices are moving or the interference is high, the standard deviation is high. For Figs. 3a–d we used the Device Analyzer dataset while for Fig. 3e we conducted the experiments by ourselves.

Figure 3 provides evidence that most of the time, mobile devices are accessible via Bluetooth or WiFi and can efficiently exchange data.

Answer: A neighbor can be helpful by enabling the Bluetooth, the WiFi, and WiFi-direct interfaces to be able to receive tasks for execution. It the case of D2D offloading, a mobile user can be helpful whenever they have the Bluetooth interface ON, but are not connected to another device, or whenever they have the WiFi-direct interface ON, and their device is detectable. It is worth mentioning that depending on the version of the operating system of the mobile device, WiFi-direct may not be able to work in parallel with the default WiFi, which means that a device can either be connected to an access point or to another device. However, it is easy to automate the process of enabling the WiFi-direct interface whenever WiFi is ON, but it is not connected to an access point. In the case of two hop offloading that is assisted by an access point, a neighbor can be helpful at any time.

Question 3: When Is a Neighbor Able to Help?

Given that mobile devices have available resources and that there exist various ways to access them as we discussed in the previous two sections, in this section we examine the conditions under which mobile devices coexist in a location and can be connected. For that reason, we employ a mobile big data dataset that contains information regarding the mobility of 16,720 mobile users that are browsing in Shanghai, and while moving, they were connected to one of the 2651 cellular access points. Figure 4 shows the popularity of each cellular access point and the mobility of the participants. The y-axis in both plots is in logarithmic scale. The popularity between the cellular access points varies significantly since some access points are located in popular areas from where many users are passing while there exist access points with

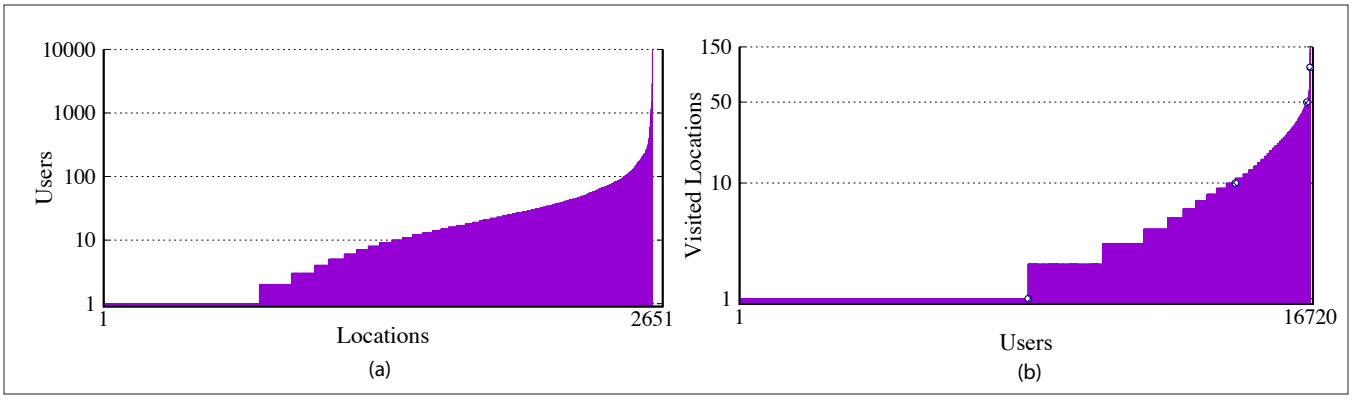

Figure 4. Mobility of users in Shanghai and popularity of each cellular access point.

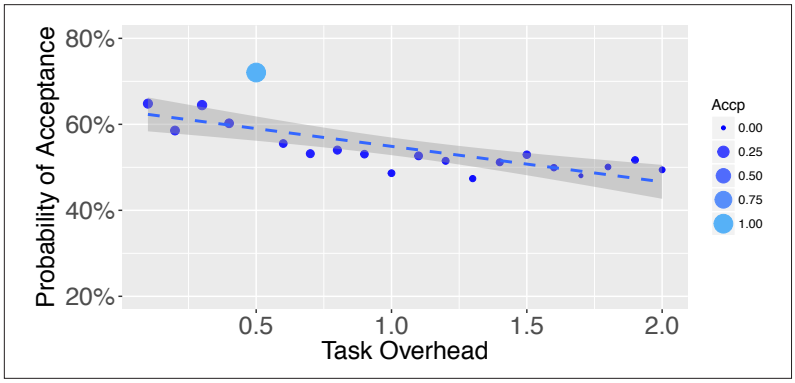

Figure 5. The probability of accepting an offloaded task based on the task overhead in terms of battery drop.

less popularity. Cellular access points with high popularity can host offloading mechanisms and collect information that can be useful in the selection of the most suitable users for offloading. For example, they can keep track of the time the users stay connected, or they can ask the users to report their current resource availability. Moreover, information regarding cellular access point popularity can be used by mobile users individually on the selection of the proper time to ask for help in cases where the offloadable applications do not have time constraints.

Answer: Most of the time mobile users are surrounded by other mobile users who have available resources and can potentially connect with them. Also, edge devices and smart routers in popular locations can collect data regarding users' mobility and provide suggestions on the neighbor selection process.

Question 4: Why Would a Neighbor Be Willing to Help?

Although there exist nearby devices that have available resources and there are ways to contact them and offload tasks to them, it is still not clear why the owners of these devices should share their resources for unknowns. We have surveyed 733 users where we asked them whether they are willing to execute an offloaded task that has a task overhead, in the sense of battery drop, between 0 and 2 percent. Figure 5 shows the response of the users where the probability of a user to accept the task is more than 50 percent.

There are two main ways to incentivize mobile users to share their resources: either by paying them with some credits that can be used when they are using the system or they can exchange with something else [5]; or by assigning each user a reputation score that affects the quality of the services they have accessed.

Answer: Because they are motivated by the rewards or because they are altruists. Hidden market design offers a range of tools that allow users to select the fraction of their resources they are willing to share and under which conditions they want to share them. For example, "whenever the battery level is more than 50 percent, and the CPU utilization is less than 50 percent, accept an offloaded task and create a virtual machine with a prespecified set of characteristics to execute the task".

Question 5: Is it Safe to Execute My Neighbor's Task?

Since the use of wireless sensor networks, security and privacy are two main concerns to maintain a safe and trusted ecosystem [12]. Researchers have developed novel cryptographic techniques that support constrained devices and keep the transmission secure. Public key authentication schemes (RSA), elliptic-curves (ECC), and symmetric encryption are methods to ensure a secure channel. However, these methods still rely on an external certificate authority (CA) that provides a secure and trusted channel to share the keys that need to be used in the transmission between peers (i.e., the same key for symmetric encryption, a master key for elliptic-curve approaches).

Furthermore, there is also a need to ensure the data/task integrity stored in a neighbor's device. Therefore, the data stored in others' devices have to be encrypted in order to keep the privacy intact [13, 14]. Moreover, sensible data needs to be obfuscated before the offloading process to another's device, so that the nearby device will still be able to perform the task efficiently and accurately. Existing cryptographic approaches provide methods such as public key schemes where the neighbor's device can authenticate the origin of the data/task but cannot decrypt the sensible content. These techniques allow the devices to perform the offloaded task. Homomorphic encryptions can be used in these mobile offloading scenarios [15]. With these algorithms one can compute any arbitrary mathematical compu-

tations on encrypted data which can be reduced to a composition of addition and multiplication gates, although some of them are less resilient to attacks but offer better performance in computational terms. The encryption of these tasks needs to be fast (i.e., computationally efficient encryption-decryption algorithms) and its overhead must be minimal in order to achieve fast offloading transmission, and not overload the neighbor's device resources such as memory.

Answer: Yes. From the helpers point of view, the shared resources are utilized in an isolated and virtualized environment that does not have access to locally stored data. In case the offloading task tries to gain access to extra resources or it does not terminate as expected, it can just be deleted. From the offloading device point of view, cryptographic techniques, as explained above, allow the task to be executed without giving access to the original data.

Conclusion

We used five questions to communicate the feasibility of task offloading with the help of two datasets that we have accessed and one study we conducted. We conclude that mobile devices have available resources, are accessible via various types of communication protocols, are nearby, are willing or motivated to help, and it is safe for them and for the device that asks for help.

We conclude that mobile devices have available resources, are accessible via various types of communication protocols, are nearby, are willing or motivated to help, and it is safe for them and for the device that asks for help.

Acknowledgement

This research has been supported in part by projects 26211515 and 16214817 from the Research Grants Council of Hong Kong.

References

[1] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 white paper," http://www.cisco.com/c/en/us/solutions/serviceprovider/ visual-networking-index-vni/index.html, Mar. 2017.

[2] S. Kosta *et al.*, "Thinkair: Dynamic Resource Allocation and Parallel Execution in the Cloud for Mobile Code Offload ing," *Proc. IEEE INFOCOM*, 2012.

[3] E. Cuervo *et al.*, "Maui: Making Smartphones Last Longer with Code Offload," *Proc. 8th Int'l. Conf. Mobile Systems, Applications, and Services*, ser. MobiSys '10, 2010, pp. 49–62.

[4] C. Shi *et al.*, "Serendipity: Enabling Remote Computing Among Intermittently Connected Mobile Devices," *Proc. MobiHoc '12*, 2012, pp. 145–54.

[5] D. Chatzopoulos *et al.*, "Have You Asked Your Neighbors? A Hidden Market Approach for Device-to-Device Offloading," *Proc. 2016 IEEE 17th Int'l. Symposium A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2016, pp. 1–9.

[6] M. Condoluci *et al.*, "LTE-Direct vs. Wifi-Direct for Machine-Type Communications over LTE-A Systems," *Proc. 2015 IEEE 26th Annual Int'l. Symposium Personal, Indoor, and Mobile Radio Communications (PIMRC)*, IEEE, 2015, pp. 2298–2302.

[7] D. Chatzopoulos *et al.*, "Video Compression in the Neighborhood: An Opportunistic Approach," *Proc. IEEE ICC 2016 Ad-hoc and Sensor Networking Symposium (ICC'16 AHSN)*, Kuala Lumpur, Malaysia, May 2016.

[8] M. Musolesi, "Big Mobile Data Mining: Good or Evil?" *IEEE Internet Computing*, vol. 18, no. 1, pp. 78–81, Jan. 2014; available: http://dx.doi.org/10.1109/MIC.2014.2

[9] F. Boccardi *et al.*, "Five Disruptive Technology Directions for 5G," *IEEE Commun. Mag.*, vol. 52, no. 2, 2014, pp. 74–80.

[10] D. T. Wagner, A. Rice, and A. R. Beresford, "Device Analyzer," *Proc. HOTMOBILE 2011 12th Workshop on Mobile Computing Systems and Applications*, 2011.

[11] C. Bermejo, D. Chatzopoulos, and P. Hui, "How Sustainable is Social Based Mobile Crowdsensing? An Experimental Study," *Proc. 2016 IEEE 24th Int'l. Conf. Network Protocols (ICNP)*, Nov. 2016, pp. 1–6.

[12] X. Chen *et al.*, "Sensor Network Security: A Survey," *IEEE Commun. Surveys & Tutorials*, vol. 11, no. 2, 2009.

[13] S. Bajaj and R. Sion, "Trusteddb: A Trusted Hardware-Based Database with Privacy and Data Confidentiality," *IEEE Trans. Knowledge and Data Engineering*, vol. 26, no. 3, 2014, pp. 752–65.

[14] F. Y.-F. Wang, "Cryptographically Enforced Access Control for User Data in Untrusted Clouds," Ph.D. dissertation, Massachusetts Institute of Technology, 2016.

[15] C. Gentry *et al.*, "Fully Homomorphic Encryption Using Ideal Lattices," *Proc. STOC*, vol. 9, no. 2009, 2009, pp. 169–78.

Biographies

Dimitris Chatzopoulos (dcab@cse.ust.hk) received his Diploma and his M.Sc. in computer engineering and communications from the Department of Electrical and Computer Engineering, University of Thessaly, Volos, Greece. He is currently a Ph.D. student in the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, and a member of the HKUST-DT System and Media Lab. His main research interests are in the areas of mobile computing, device-to-device ecosystems and cryptocurrencies.

Carlos Bermejo (cbf@connect.ust.hk) received the M.Sc. degree in telecommunication engineering from Oviedo University, Spain, in 2012. He worked for two years at Telekom Innovation Laboratories (T-Labs), and as a software developer for a startup company (Germany). He is currently pursuing the Ph.D. degree at the Hong Kong University of Science and Technology. He is with the Symlab Research Group. His primary research interests are Internet of Things, privacy, mobile augmented reality, human-computer-interaction, and device-to-device communications.

Ehsan ul Haq (euhaq@connect.ust.hk) received the M.Sc. degree in information technology and sciences from Skolkovo Institute of Science and Technology, Russia, in 2015. He was also a visiting research student at Massachusetts Institute of Technology, USA. He is currently pursuing the Ph.D. degree at The Hong Kong University of Science and Technology and is member of the HKUST-DT System and Media Lab. His main research interests are data analytics, user behavior analysis, computational politics, and human-computer-interaction.

Yong Li (liyong07@tsinghua.edu.cn) received the B.S. degree in electronics and information engineering from Huazhong University of Science and Technology, Wuhan, China, in 2007, and the Ph.D. degree in electronic engineering from Tsinghua University, Beijing, China, in 2012. He is currently a faculty member of the Department of Electronic Engineering, Tsinghua University. He received the IEEE 2016 ComSoc Asia-Pacific Outstanding Young Researchers and Young Talent Program of China Association for Science and Technology.

Pan Hui (panhui@cse.ust.hk) received his Ph.D. degree from the Computer Laboratory, University of Cambridge, and his M.Phil. and B.Eng. from the University of Hong Kong. He is the Nokia Chair Professor in Data Science at the University of Helsinki and director of the HKUST-DT System and Media Lab at the Hong Kong University of Science and Technology. He is an IEEE Fellow and an ACM Distinguished Scientist.