Crowdsourcing on Mobile Cloud: Cost Minimization of Joint Data Acquisition and Processing

Huan Ke, Peng Li and Song Guo
School of Computer Science and Engineering
The University of Aizu, Aizu-Wakamatsu, Japan
{m5172105,pengli, sguo}@u-aizu.ac.jp

***Abstract*—As the advance of mobile devices, crowdsourcing has been successfully applied in many scenarios by employing distributed mobile devices to collectively monitor a diverse range of human activities and surrounding environment. Unfortunately, treating mobile devices as simple sensors that generate raw sensing data may lead to low efficiency because of excessive bandwidth occupation and additional computation resource consumption. In this paper, we integrate crowdsourcing into existing mobile cloud framework such that data acquisition and processing can be conducted in a uniform platform. We consider a dynamic network where mobile devices may join and leave the network at any time. To deal with the challenges of sensing and computation task assignment in such a dynamic environment, we propose an online algorithm with the objective of minimizing the total cost including sensing, processing, communication and delay cost. Extensive simulations are conducted to demonstrate that the proposed algorithm can significantly reduce the total cost of crowdsourcing.**

I. Introduction

Modern mobile devices, e.g., smartphones and tablets, are equipped with a set of powerful embedded sensors, such as accelerometer, GPS, microphone, and camera. The existence of a huge number of such mobile devices motivates the idea of crowdsourcing [1]–[5] that employs distributed mobile devices to collectively monitor human activities and surrounding environment, without the need of deploying thousands of dedicated sensors. A typical process of crowdsourcing includes three stages. First, a control center assigns sensing tasks to mobile devices. Then, mobile devices conduct sensing tasks individually. Finally, sensing data are sent back to the control center for further processing.

Although crowdsourcing has succeeded in many applications, such as Nericell [6], iStockphoto [7] and VTrack [8], it suffers from two weaknesses that would lead to low efficiency. First, collection of raw sensing data would occupy excessive network bandwidth. Even though cellular network bandwidth has been significantly improved by modern cellular technologies, such as 4G, it is still a kind of scarce resource because a large number of mobile devices may submit their sensing data simultaneously. Furthermore, crowdsourcing applications may share the network bandwidth with other bandwidth-hungry applications, such as video streaming [9]. Second, additional computation resources, e.g., workstations or cloud, are needed to process the collected sensing data. The separation of data acquisition and processing would incur high system complexity and large latency for end users.

Mobile cloud computing is a promising paradigm by enabling mobile devices to work collaboratively as cloud resource providers [10]–[14]. In contrast to traditional mobile cloud in which mobile devices provide only computation resources, we integrate crowdsourcing into existing mobile cloud such that data acquisition and processing can be conducted in a uniform platform. Since the sensing data can be processed by local mobile devices, the centralized computation resources in traditional crowdsourcing platform can be minimized, or even completely eliminated. Furthermore, the data size after processing may be significantly reduced to save network bandwidth. To achieve efficient crowdsourcing on mobile cloud, we need to deal with the following challenges. First, mobile devices are diverse in sensing functions, storage, and computation capability. Therefore, we need to optimize mobile resource utilization via appropriate assigning sensing and computation tasks among mobile devices. Second, the mobile cloud is a dynamic system, where mobile devices may join and leave the cloud at any time. By conquering all these challenges, we summarize our main contributions as follows.

- First, we propose a novel platform by integrating crowdsourcing and mobile cloud computing, which can save network bandwidth and reduce hardware investment.
- Second, we consider a dynamic mobile cloud environment, and propose an online algorithm to assign crowdsourcing jobs to mobile devices with the objective of minimizing the total cost including sensing, computation, communication and delay cost.
- Finally, extensive simulations are conducted to show the advantages of our proposals.

The rest of the paper are organized as follows. Related work is reviewed in Section II. Section III presents the system model. An online algorithm is proposed in Section IV. Section V presents simulation results. We conclude the paper in Section VI.

II. Related work

Recently, mobile cloud computing has emerged as an extension of cloud computing. Luo et al. [15] have introduced the idea of using cloud computing to enhance the capabilities

978-1-4799-3088-3/14/$31.00 ©2014 IEEE

of mobile devices. Marinelli et al. [16] have proposed Hyrax, a mobile cloud computing that allows mobile devices to use cloud computing platforms for data processing. Specifically, Oberheide et al. [17] have presented a system that outsources antivirus services from mobile devices to the cloud. Mobile cloud has become a service model that allows mobile devices to use the computing and storage resources provided by cloud without complex hardware and software implementations at mobile devices. However, offloading the mobile applications onto the cloud would occupy excessive network bandwidth, leading to high communication cost. Thus, a partition scheme has emerged to offload parts of an application to cloud for a better performance. CloneCloud [18] is a flexible application partitioner and execution runtime in an application-level virtual machine to seamlessly offload parts of their execution from devices to cloud. Lei et al. [19] aim at optimizing the partition of a data stream application between the mobile devices and the cloud such that the application has maximum throughout in processing the stream data. Nkosi et.al [20] have proposed multimedia and security operations that can be performed in the cloud, allowing mobile health service providers to subscribe and extend the capabilities of their mobile health applications beyond the existing mobile device limitations.

Above work depends on resources provided by third-party cloud computing platforms [21]–[24]. However, these resources are not always available because of poor network connection. To overcome this issue, a novel mobile cloud computing platform using mobile devices as computing resources has been proposed. GonzaIo et al. [10] have established a mobile cloud by exploiting a collection of nearby mobile devices. The framework takes advantages of pervasiveness of mobile devices by allowing them to execute jobs on mobile devices. In order to form such a mobile cloud, each mobile device needs to contribute its own resources for data processing. Due to a large amount of resources provided by multiple mobile devices, resource management has emerged as a vital issue for mobile cloud computing. Previous work [25]–[28] has explored the resource management in traditional mobile cloud computing. They take not only the radio resources for wireless access but also the computing resource for data processing into consideration. However, the resource management schemes in the literature do not consider to integrate data acquisition and processing as we do in this paper.

III. System Model

We consider a discrete-time model for a dynamic mobile cloud consisting of a set $N(t)$ of mobile devices in time slot t, as shown in Fig. 1. Each device $n_i \in N(t)$ can conduct at most S_i sensing tasks in each time slot due to the constraints of sensing hardware. The computation capability of device $n_i \in N(t)$ is denoted by P_i.

The crowdsourcing jobs arrive in an online manner, and the set of jobs in time slot t is denoted by $J(t) = \{j_1, j_2, \cdots, j_{|J(t)|}\}$. Each job j_i arriving at time τ_i consists of two sub-tasks, sensing and processing, that can be conducted

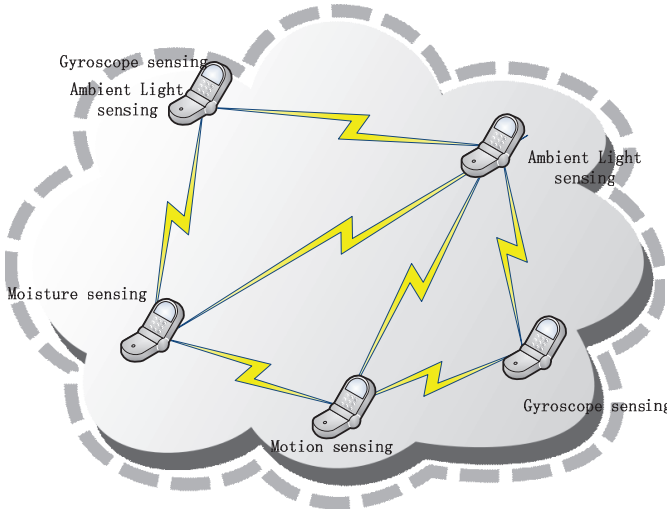

Fig. 1. The mobile cloud.

sequentially by one device or two different ones. Each job $j_i \in J$ needs to consume p_i computational resource units. We assume that both sensing and processing tasks of a job can be finished within a time slot.

To model the sensing task assignment, we define a binary variable x_i^k as follows:

$$x_i^k = \begin{cases} 1, \text{ if sensing task of job } j_k \text{ is assigned to } n_i, \\ 0, \text{ otherwise.} \end{cases} \quad (1)$$

At each time slot t, at most S_i sensing tasks can be assigned to device n_i, i.e.,

$$\sum_{j_k \in J(t)} x_i^k \leq S_i, \forall n_i \in N(t). \quad (2)$$

Note that sensing and processing tasks of each job may be conducted by different mobile devices. We define a binary variable y_i^k to denote whether the processing task of job j_k is assigned to device n_i, i.e.,

$$y_i^k = \begin{cases} 1, \text{ if processing task of job } j_k \text{ is assigned to } n_i, \\ 0, \text{ otherwise.} \end{cases} \quad (3)$$

The set of processing tasks assigned to device n_i is constrained by its computation capability, i.e.,

$$\sum_{j_k \in J(t)} y_i^k p_k \leq P_i, \forall n_i \in N(t). \quad (4)$$

We consider four kinds of cost in the mobile cloud: storage, computation, communication and job delay, which are elaborated as follows.

Sensing cost: We let a_k denote the sensing cost of job j_k, thus the total sensing cost $C_{sen}(t)$ can be calculated by:

$$C_{sen}(t) = \sum_{j_k \in J(t)} \sum_{n_i \in N(t)} a_k x_i^k. \quad (5)$$

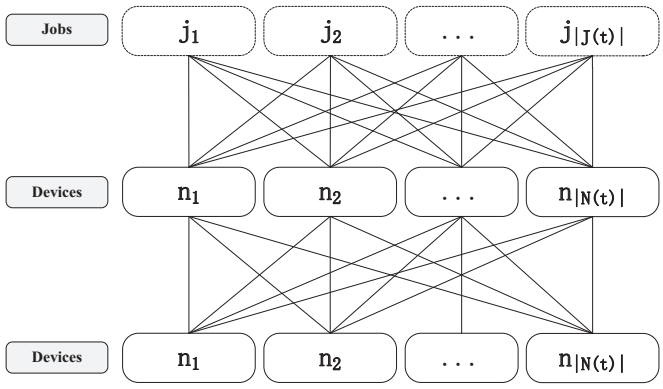

Fig. 2. The tuples of all possible assignments.

Computation cost: We let b_i denote the cost of unit computation resource at n_i, and the total computation cost $C_{comp}(t)$ is:

$$C_{comp}(t) = \sum_{j_k \in J(t)} \sum_{n_i \in N(t)} b_i p_k y_i^k. \quad (6)$$

Communication cost: If the sensing and processing tasks of a job are conducted by different devices, a communication cost is incurred because of data delivery between two devices. Otherwise, no communication cost is incurred. We define β to denote the communication cost of unit data, and the total communication cost can be expressed by:

$$C_{comm}(t) = \beta \sum_{j_k \in J(t)} \sum_{n_i, n_{i'} \in N(t)}^{i \neq i'} x_i^k y_{i'}^k e_k, \quad (7)$$

where e_k is the size of the sensing data.

Delay cost: The delay cost of each job n_i can be calculated by a function $F_i(d_i - \tau_i)$, where d_i is the finish time of job n_i. Note that the function $F_i(x)$ is determined by the characteristic of the job. For example, it can be defined as an exponential function for real-time jobs. The total delay cost of jobs finished in time t is calculated by:

$$C_{delay}(t) = \sum_{j_k \in J(t)} \Big(F_i(t - \tau_i) \sum_{n_i \in N(t)} x_i^k \sum_{n_i \in N(t)} y_i^k \Big). \quad (8)$$

The total cost within a long time period $[1, T]$ can be calculated by:

$$C_{total} = \sum_{t=1}^{T} (C_{sen}(t) + C_{comp}(t) + C_{comm}(t) + C_{delay}(t)). \quad (9)$$

Our objective is to minimize the total cost C_{total}, which is determined by the assignment of both sensing and processing tasks. In the next section, we develop an efficient online algorithm that jointly considers both data acquisition and processing.

IV. Algorithm design

In this section, we design an online algorithm called JDAP (Joint Data Acquisition and Processing) to minimize the total cost of crowdsourcing jobs on mobile cloud by jointly considering data acquisition and processing. Our basic idea is to iteratively assign crowdsourcing jobs to mobile devices to minimize the total cost in a greedy manner. The pseudo codes of our proposed algorithm are shown in the following Algorithm 1.

Algorithm 1 The JDAP Algorithm

Input: $J(t), N(t)$;
Output: Job assignment in the form of x_i^k and y_i^k;

```
S_i = S_i, ∀n_i ∈ N(t);
P_i = P_i, ∀n_i ∈ N(t);
create a set of tuples M(t) = J(t) × N(t) × N(t);
partition the jobs in J(t) into several subsets {J_0(t), J_1(t), ...J_z(t)} according to their arriving time;
for l = 0 to z do
    for r = 1 to |J_l(t)| do
        find a tuple ⟨j_k, n_i, n_i'⟩, j_k ∈ J_l(t) leading to the minimum total cost in current step;
        if S_i − 1 ≥ 0 and P_i − p_k ≥ 0 then
            assign sensing task of job j_k to node n_i by letting x_i^k = 1;
            assign processing task of job j_k to node n_i' by letting y_i'^k = 1;
            S_i = S_i − 1;
            P_i' = P_i' − p_k;
        end if
    end for
end for
```

Our proposed online algorithm will be executed in the beginning of each time slot. With input of current job set $J(t)$ and device set $N(t)$, it generates task assignment decisions of x_i^k and y_i^k. We maintain two variables $\mathcal{S}_i$ and $\mathcal{P}_i$ to indicate the residual sensing and processing capability of each device $n_i \in N(t)$, which are initialized to be S_i and P_i, respectively. Then, we create a set of tuples $M(t) = J(t) \times N(t) \times N(t)$, each of which denotes a possible job assignment as illustrated in Fig. 2. We partition the jobs in $J(t)$ into several subsets according to their lateness. For example, the jobs in $J(t)$ with the earliest arriving time are stored in $J_0(t)$, while the ones coming in current time slot are maintained in $J_z(t)$. After that, we iteratively assign jobs to mobile cloud by starting from set $J_0(t)$. In each iteration, we find a tuple $\langle j_k, n_i, n_{i'} \rangle, j_k \in J_l(t)$ leading to the minimum total cost in current step. If the residual sensing and processing capability is enough to accommodate this job, we assign j_k to sensing node n_i and processing node $n_{i'}$. We finish the current iteration by updating the values of both S_i and P_i. In our proposed algorithm, the jobs arriving earlier always have higher priority in job assignment, which could avoid large delay cost.

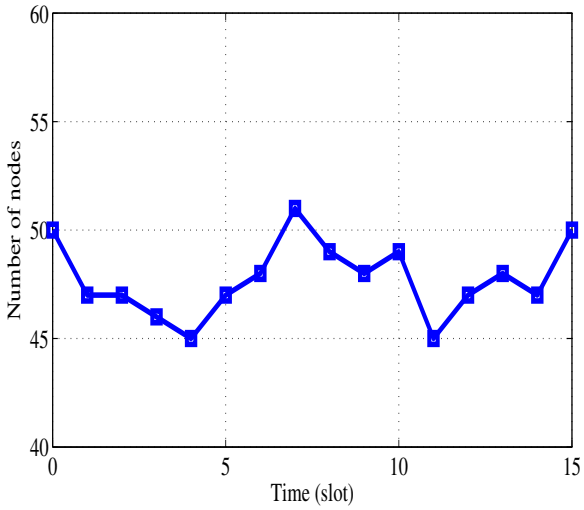

Fig. 3. The number of nodes in each time slot.

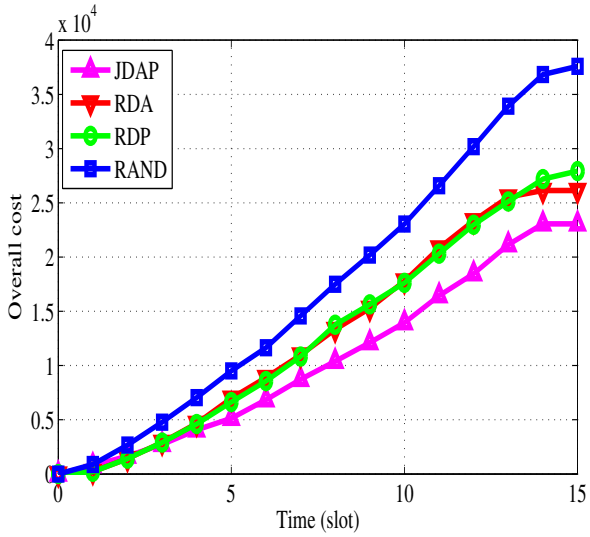

Fig. 4. The Overall cost at different slots

V. Performance Evaluation

In this section, we conduct extensive simulations to evaluate the performance of our proposed algorithm. For comparison, we also consider other three algorithms:

- RDA (Random Data Acquisition): an algorithm that considers only processing cost during job assignment.
- RDP (Random Data Processing): an algorithm that considers only data acquisition cost during job assignment.
- RAND (Random algorithm): a random job assignment algorithm.

We first consider a dynamic mobile cloud that initially contains 50 nodes. In each time slot, there are some nodes leaving or joining the mobile cloud, and the number of nodes in each time slot is shown in Fig. 3. The number of sensing tasks that can be accommodated by each node is randomly specified as a uniform distribution between 4 and 8, and computational resources are randomly distributed within [5, 10]. We consider a total number of 50 jobs that arrive in an online manner, and the results of all algorithms are shown in Fig. 4. Although the overall cost of all algorithms increases as time progresses, our proposed JDAP algorithm always outperforms other algorithms, and their performance gap becomes larger. For example, the RAND algorithm incurs 1.66 times more cost than JDAP at the 15-th time slot.

We then study the influence of sensing capability of mobile devices on the total cost. We consider a similar dynamic mobile cloud and a set of online jobs with the previous simulation. The sensing capability of mobile devices is randomly specified according to a uniform distribution. We show the results under three sensing capability ranges, [2,6], [6,10] and [10,14], in Fig. 5. All results are averaged over 50 random network instances. As sensing capability increases, the overall cost of all algorithms decreases because more jobs can be conducted by the mobile cloud in each time slot, leading to a reduced delay cost. Moreover, the advantages of our proposed

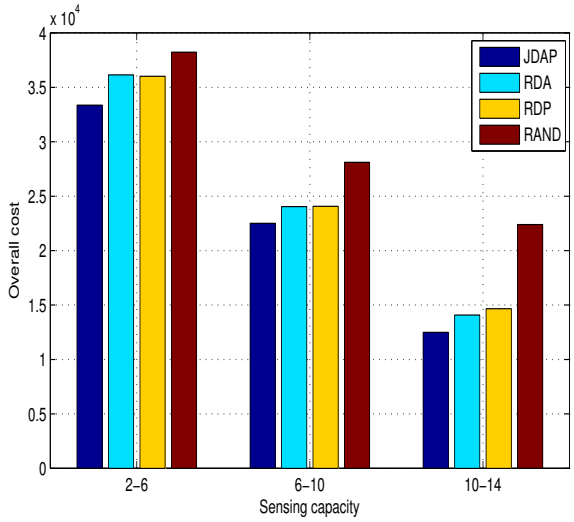

Fig. 5. The overall cost versus different sensing capability.

algorithm are more obvious under larger sensing capability due to its joint consideration of both data acquisition and processing.

Finally, we investigate the performance of our proposed algorithm under different processing capability that is uniformly distributed within three ranges: [3,7], [7-11], and [11-15]. As shown in Fig. 6, the performance of all algorithms improves as the growth of processing capability because of a similar reason in last set of simulations. Furthermore, our proposed algorithm JDAP outperforms other algorithms under all settings.

VI. Conclusion

In this paper we propose to integrate crowdsourcing into mobile cloud. Without offloading the applications to the cloud, the mobile cloud can use the resources provided by multiple mobile devices to complete the applications. We design an

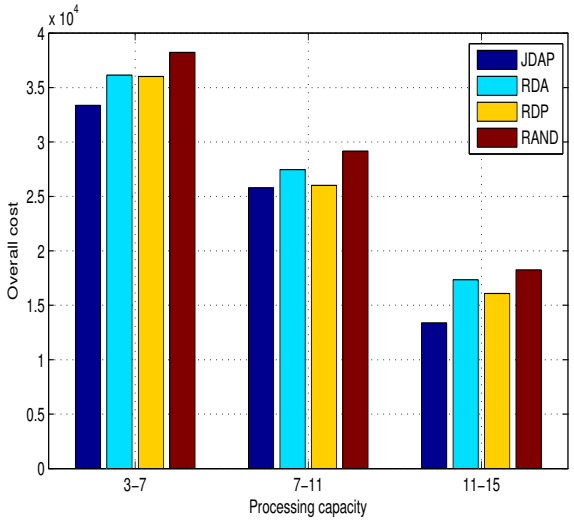

Fig. 6. The overall cost versus different processing capability.

online algorithm to determine the assignment of data acquisition and processing tasks with the objective of minimizing the overall cost including sensing, computation, communication and delay. Finally, we conduct extensive simulations to show that our proposals outperform other algorithms based on random assignment.

REFERENCES

[1] G. Chatzimilioudis and D. Zeinalipour-Yazti, "Crowdsourcing for mobile data management," in *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, vol. 2. IEEE, 2013, pp. 3–4.

[2] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, "Crowdsourcing with smartphones," *Internet Computing, IEEE*, vol. 16, no. 5, pp. 36–44, 2012.

[3] M. Stevens and E. DHondt, "Crowdsourcing of pollution data using smartphones," in *Workshop on Ubiquitous Crowdsourcing*, 2010.

[4] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner, "mcrowd: a platform for mobile crowdsourcing," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2009, pp. 347–348.

[5] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing," in *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 2012, pp. 173–184.

[6] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*. ACM, 2008, pp. 323–336.

[7] D. C. Brabham, "Moving the crowd at istockphoto: The composition of the crowd and motivations for participation in a crowdsourcing application," *First Monday*, vol. 13, no. 6, 2008.

[8] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2009, pp. 85–98.

[9] Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-peer Networking and Applications*, vol. 1, no. 1, pp. 18–28, 2008.

[10] G. Huerta-Canepa and D. Lee, "A virtual cloud computing provider for mobile devices," in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. ACM, 2010, p. 6.

[11] D. Huang *et al.*, "Mobile cloud computing," *IEEE COMSOC Multimedia Communications Technical Committee (MMTC) E-Letter*, vol. 6, no. 10, pp. 27–31, 2011.

[12] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, 2011.

[13] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84–106, 2013.

[14] A. Klein, C. Mannweiler, J. Schneider, and H. D. Schotten, "Access schemes for mobile cloud computing," in *Mobile Data Management (MDM), 2010 Eleventh International Conference on*. IEEE, 2010, pp. 387–392.

[15] X. Luo, "From augmented reality to augmented computing: a look at cloud-mobile convergence," in *Ubiquitous Virtual Reality, 2009. ISUVR'09. International Symposium on*. IEEE, 2009, pp. 29–32.

[16] E. E. Marinelli, "Hyrax: cloud computing on mobile devices using mapreduce," DTIC Document, Tech. Rep., 2009.

[17] J. Oberheide, K. Veeraraghavan, E. Cooke, J. Flinn, and F. Jahanian, "Virtualized in-cloud security services for mobile devices," in *Proceedings of the First Workshop on Virtualization in Mobile Computing*. ACM, 2008, pp. 31–35.

[18] 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*. ACM, 2011, pp. 301–314.

[19] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, and A. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 4, pp. 23–32, 2013.

[20] M. Nkosi and F. Mekuria, "Cloud computing for enhanced mobile health applications," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*. IEEE, 2010, pp. 629–633.

[21] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation computer systems*, vol. 25, no. 6, pp. 599–616, 2009.

[22] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[23] L. Wang, G. Von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud computing: a perspective study," *New Generation Computing*, vol. 28, no. 2, pp. 137–146, 2010.

[24] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The eucalyptus open-source cloud-computing system," in *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*. IEEE, 2009, pp. 124–131.

[25] R. Kaewpuang, D. Niyato, P. Wang, and E. Hossain, "A framework for cooperative resource management in mobile cloud computing," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 12, pp. 2685–2700, 2013.

[26] H. Liang, D. Huang, L. X. Cai, X. Shen, and D. Peng, "Resource allocation for security services in mobile cloud computing," in *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*. IEEE, 2011, pp. 191–195.

[27] X. Jin and Y.-K. Kwok, "Cloud assisted p2p media streaming for bandwidth constrained mobile subscribers," in *Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on*. IEEE, 2010, pp. 800–805.

[28] E. Jung, Y. Wang, I. Prilepov, F. Maker, X. Liu, and V. Akella, "User-profile-driven collaborative bandwidth sharing on mobile phones," in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. ACM, 2010, p. 2.