Procedia Computer Science

Volume 80, 2016, Pages 1232–1241

ICCS 2016. The International Conference on Computational Science

A Heuristic Algorithm for Multi-Site Computation Offloading in Mobile Cloud Computing

Nur Idawati Md Enzai[1]*and Maolin Tang[2]†

[1] nuridawati.mdenzai@hdr.qut.edu.au
[2] m.tang@qut.edu.au

Abstract

Due to limitation of mobile device in terms of battery life and processing power, Mobile Cloud Computing (MCC) has become an attractive choice to leverage this shortcoming as the mobile computation could be offloaded to the cloud, which is so-called *mobile computation offloading*. Existing research on mobile computation offloading considers offloading a mobile computation to a single cloud. However, in the real world a computation service could be provided by multiple clouds and each computation service. Thus, a new and interesting research problem in mobile computation offloading is how to select a computation service for each of the computation tasks of a mobile computation such that the computation time of the mobile computation, the energy consumption of the mobile device and the cost of using the computation services are minimized. This is so called multi-site computation offloading in mobile cloud computing. In this paper we formulate the multi-site computation offloading problem, propose a heuristic algorithm for the multi-site computation offloading problem and evaluate the heuristic algorithm.

Keywords: Computation offloading, mobile cloud computing, scheduling, heuristic algorithm

1 Introduction

Cloud computing provides services to clients in forms of processing and storage without the need for clients to install hardware on their side. As the number of mobile users increases, the concept of Mobile Cloud Computing (MCC) emerges. Dinh, Lee, Niyato, and Wang define MCC as the cloud provisioning of data processing and storage services for mobile users [6]. High processing speed and powerful memory capacity of mobile device are not essential because clouds can process all the complicated computing modules. Therefore, mobile device can take advantage of cloud services to perform large amounts of computation (instructions), which is so called

*N. I. Md Enzai is with School of Electrical Engineering and Computer Science, Queensland University of Technology, 2 George Street, Brisbane, Australia and Faculty of Electrical Engineering, Universiti Teknologi Mara (UiTM), 23000 Dungun Terengganu, Malaysia

†M. Tang is with School of Electrical Engineering and Computer Science, Queensland University of Technology, 2 George Street, Brisbane, Australia

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2016

doi:10.1016/j.procs.2016.05.490

Computation Offloading. Kumar and Lu prove that computation offloading is beneficial with computation-intensive tasks [10].

To cloud improve the performance of a mobile computation, the tasks of the computation could be offloaded to multiple clouds. A main motivation for multi-cloud is the ability to offer different prices at different performances such as computation time [11]. Moreover, application designers may aim to achieve different performance objectives (e.g. throughput, reliability, cost). This can be achieved by utilizing the resources in cloud providers that have different performance capacities and charged prices. Multi-cloud resource allocation also benefits from the best combination of computation services from multi-cloud providers [14].

To our knowledge, most computation offloading works only consider one single cloud provider [4, 5, 8, 9]. Wu and Huang proposed mobile cloud service composition and Heo, Kim, and Suh distribute task sharing to multiple clouds to reduce delay for online gaming, but both works do not address scheduling problem [7, 15]. Meanwhile other multi-cloud resource allocation and scheduling works do not consider mobile device in the problem [3, 2, 13, 14]. Our work also takes into account multiple objectives instead of dealing with them separately.

Even though multi-site computation offloading has been addressed, only one cloud provider is considered. Even if multi-cloud is considered, only the assignment or mapping part is covered. Whereas, in the case of computational tasks workflow, the tasks need to be scheduled as well. Current multi-cloud offloading works also do not address energy, computation time and price simultaneously.

We aim to assign the workflow of computational tasks to services provided by clouds or mobile device as well as schedule them while minimizing overall mobile user requirements namely energy, completion time and price. From the computational point of view, computational tasks workflow assignment and scheduling problem is a typical constrained combinatorial optimization problem. Even though clouds are assumed to always be able to cater the execution of tasks due to its multi-tenancy features, mobile device on the other hand is assumed to be able to handle only a particular number of tasks at a time.

An example of MCC application scenario is when a mobile user travels to foreign country and lost his or her way. As Global Positioning System (GPS) alone may not be enough, the user may capture short video or images and send them to the cloud to be processed to obtain information of the whereabouts. This involves high processing power to extract features and match with large repository [16]. The processed data will give the user some information on his or her location. The user may also want to make use of social media for example automatic blogging as addressed in [12] and translation services as provided by [1]. However, a mobile user is constrained in terms of mobile device battery life, timing and monetary budget. Instead of relying to a single cloud provider, a mobile user may include more cloud providers with more services and varying capacities to choose from.

Since this is our preliminary research on the problem, we propose a heuristic algorithm to further improve the quality of solutions. Different from the existing multi-site computation offloading algorithm, this new heuristic algorithm also schedules the computational tasks at cloud providers and mobile device as well as addresses three criteria namely energy consumption of mobile device, tasks completion time and charged price at the same time.

The remainder of the paper is organized as follows. First of all, we formulate the research problem in Section 2. After that, we present our heuristic algorithm and evaluation results in Section 3 and Section 4, respectively. This research is finally concluded and future work is discussed in Section 5.

2 Problem Formulation

Our goal is to assign all the computation tasks in the workflow of a mobile computation to either a cloud that provides the corresponding computation service or the mobile device and to schedule the computation tasks assigned to the clouds and mobile device such that the following three objectives are minimized simultaneously: the energy consumption of the mobile device, the total computation time, and the cloud computing cost incurred by the computation offloading. Figure 2 is an instance of the workflow of a mobile computation which has 10 computation tasks and the control dependencies between them.

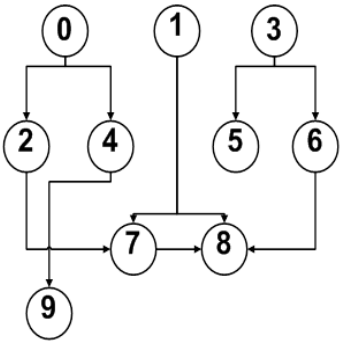

Figure 1: An instance of workflow

2.1 Inputs

1. A computation workflow is represented as a Directed Acyclic Graph (DAG): $G = (V, E)$ where $V = \{v_1, v_2, ..., v_m\}$ is a set of computation tasks, $e =< v_i, v_j > \in E$ if computation task v_i must have finished before computation task v_j can start, where $1 \leq i, j \leq m$. A computation task v_i is described as a 2-tuple, $\langle data_i, inst_i \rangle$ where $data_i$ is the data size to be processed by the computation task in megabytes and $inst_i$ is the number of instructions in the computation task in millions.

2. All available services for each of the computation tasks,
$SVC = (svc_{11}, svc_{12}, ..., svc_{1k_1}), (svc_{21}, svc_{22}, ..., svc_{2k_2}), ..., (svc_{m1}, svc_{m2}, ..., svc_{mk_m})$
where svc_{ij} represents the j^{th} service of computation task v_i and k_i is the total number of services of computation task v_i.

3. A set of computation service providers for each computation task: $P = \{p_0, p_1, ..., p_j, ..., p_n\}$ where p_0 represents the mobile device and $\{p_1, ..., p_j, ..., p_n\}$ represent the cloud providers.

4. All provided services by each provider
$SVC = (svc_{01}, svc_{02}, ..., svc_{0k_0}), (svc_{11}, svc_{12}, ..., svc_{1k_1}), ..., (svc_{n1}, svc_{n2}, ..., svc_{nk_n})$, where svc_{ij} represents the j^{th} service of provider p_i and k_i is the total number of services of provider p_i.

5. Mobile device is connected to each cloud service provider through wireless link connection. Mobile device also acts as a coordinator for communication between cloud services. A set of links between mobile device and cloud services are represented as $L = \{l_1, ..., l_j, ..., l_n\}$ where $1 \leq j \leq n$ and n represents number of cloud services.

2.2 Output

An assignment and scheduling plan X of computation tasks V such that the objective function $Obj(X)$ is maximal. Let X denote an assignment and scheduling plan for computation tasks V as shown in Eq. 1 below:

$$X = ((M_1, S_1), (M_2, S_2), ..., (M_n, S_n)) \tag{1}$$

where M_i represents the selected service for computation task v_i and S_i represents the starting time of M_i.

Equation 2 gives the definition of the objective function value for X.

$$Obj(X) = \sum_{k=1}^{3} \left(\frac{Q_k^{max} - Q_k(X)}{Q_k^{max} - Q_k^{min}} \right) * w_k \tag{2}$$

where $Q_k(X)$ is the value of X for criterion k, Q_k^{max} and Q_k^{min} represent the possible maximal and minimal values of criterion k respectively and w_k is the weight for criterion k, where $1 \leq k \leq 3$ and $w_1 + w_2 + w_3 = 1$.

2.3 Constraint

Workflow scheduling is subject to the control dependency constraint as in Equation 3 below:

$$FT(v_i) \leq ST(v_j) \tag{3}$$

where $FT(v_i)$ is the finish time of computation task v_i, $ST(v_j)$ is the start time of computation task v_j, and $< v_i, v_j > \in V$.

2.4 Criteria Values

There are two types of cost value for each criterion (energy, finishing time and price), namely computation and communication. Computation corresponds to nodes in the workflow and communication corresponds to edges in the workflow. Computation cost also includes offloading cost if a computation task is assigned to a cloud service.

1. Computation Cost for a computation task v_i at a service svc_{xy}.

   Offloading time OT_{ixy} is formulated as follows:

   $$OT_{ixy} = \frac{data_i}{bws_{xy}} \tag{4}$$

   where bws_{xy} is the bandwidth to send from mobile device to svc_{xy}.
   Therefore, computation time CT_{ixy} is formulated as follows:

   $$CT_{ixy} = \frac{inst_i}{speed_{xy}} + OT_{ixy} \tag{5}$$

   where $speed_{xy}$ is the processing speed of svc_{xy}.

   Offloading energy OE_{ixy} is formulated as follows:

   $$OE_{ixy} = OT_{ixy} * power_s \tag{6}$$

where $power_s$ represents power consumed by mobile device to send to cloud service
Therefore, computation energy CE_{ixy} is formulated as follows:

$$CE_{ixy} = power_{xy} * CT_{ixy} + OE_{ixy} \tag{7}$$

where $power_{xy} = 0$, if $1 \leq x \leq n$ with n number of providers or $power_j = 10$, if $x = 0$ because we are concerned with power consumption of mobile device only. Processing power of svc_{xy} is represented by $power_{xy}$.
Offloading price OP_{ixy} is charged by the receiving side of cloud service and is formulated as follows:

$$OP_{ixy} = \frac{data_i}{bws_{xy}} * price_{xy} \tag{8}$$

Computation Price CP_{ixy} is formulated as follows:

$$CP_{ixy} = CP_{ixy} * price_{xy} + OP_{ixy} \tag{9}$$

if $x = 0$, $price_{xy} = 0$, therefore $CP_{ixy} = 0$

2. Communication cost is incurred when there exists an edge e between computation task v_i which is assigned to service svc_{x_1y} and computation task v_j which is assigned to service svc_{x_2y} where $x_1 \neq x_2$ which means that they are assigned to different providers. The computation task v_i precedes computation task v_j.

   if $x_1 = 0$ and $1 \leq x_2 \leq n$, Transmission Time TT_{ij} is formulated as follows:

$$TT_{ij} = \frac{data_i}{bws_{x_2y}} \tag{10}$$

   Therefore, Transmission Energy TE_{ij} is formulated as follows:

$$TE_{ij} = TT_{ij} * power_s \tag{11}$$

   Meanwhile, Transmission Price TP_{ij} is formulated as follows:

$$TP_{ij} = TT_{ij} * price_{x_2} \tag{12}$$

   where $price_{xy}$ is the price charged by svc_{xy}

   If $x_2 = 0$ and $1 \leq x_1 \leq n$, Transmission Time TT_{ij} is formulated as follows:

$$TT_{ij} = \frac{data_i}{bwr_{x_1y}} \tag{13}$$

   Therefore, Transmission Energy TE_{ij} is formulated as follows:

$$TE_{ij} = TT_{ij} * power_r \tag{14}$$

   Meanwhile, Transmission Price TP_{ij} is formulated as follows: Meanwhile, Transmission Price TP_{ij} is formulated as follows:

$$TP_{ij} = TT_{ij} * price_{x_1} \tag{15}$$

If $1 \leq x_1 \leq n$ and $1 \leq x_2 \leq n$, Transmission time TT_{ij} is formulated as follows:

$$TT_{ij} = \frac{data_i}{bwr_{x_1y}} + \frac{data_i}{bws_{x_2y}} \tag{16}$$

Therefore, Transmission Energy TE_{ij} is formulated as follows:

$$TE_{ij} = (\frac{data_i}{bwr_{x_1y}}) * power_r + (\frac{data_i}{bws_{x_2y}}) * power_s \tag{17}$$

Meanwhile, Transmission Price TP_{ij} is formulated as follows:

$$TP_{ij} = (\frac{data_i}{bwr_{x_1y}}) * price_{x_1} + (\frac{data_i}{bws_{x_2y}}) * price_{x_2} \tag{18}$$

3. Total Cost for an allocation or scheduling plan, X for n number of computational tasks of a workflow G is derived from total cost for nodes (computation) and edges (communication):

   Latest Finish Time, $LFT(X)$ is formulated as follows:

$$LFT(X) \quad = \quad \max_{i \in v_{exit}} (F_i) \tag{19}$$

   where v_{exit} refers to exit computation tasks in the workflow G and F_i is the finishing time of an exit task v_i.

   Energy consumed by mobile device, $Energy(X)$ is formulated as follows:

$$Energy(X) = \sum_{v_i \in V, svc_{xy} \in SVC} CE_{ixy} + \sum_{v_i \in V, v_j \in V} TE_{ij} \tag{20}$$

   Price charged on the mobile device user, $Price(X)$ is formulated as follows:

$$Price(X) = \sum_{v_i \in V, svc_{xy} \in SVC} CP_{ixy} + \sum_{v_i \in V, v_j \in V} TP_{ij} \tag{21}$$

3 Algorithm Description

For the above formulated problem, a greedy hill climbing is developed, as presented in Algorithm 1. The algorithm is a hill-climbing one. It begins with an initial assignment of all computational tasks in a workflow at mobile device shown in Steps 1-2. All the tasks are scheduled at mobile device using critical path analysis. The objective function value for the initial

solution is calculated as in Equation 2.

Initialize a solution X;
Assign G to mobile device, p_0;
Schedule G;
Calculate STs and FTs based on Equations 22, 23and 24;
Calculate $LFT(X)$ based on Equation 19;
Calculate $Energy(X)$ based on Equation 20;
Calculate $Price(X)$ based on Equation 21;
Calculate $Obj(X)$ based on Equation 2;
forall the $v_i \in V$ **do**
    $N_i = NEIGHBOUR(v_i)$;
    Generate new assignment for v_i from N_i;
    Reschedule G;
    Recalculate STs and FTs;
    Recalculate $LFT(X)$;
    Recalculate $Energy(X)$;
    Recalculate $Price(X)$;
    Recalculate $Obj(X')$;
    Select N_i such that $Obj(X')$ is maximum;
end
if $Obj(X') > Obj(X)$ **then**
    Accept the solution changes;
    $X = X'$;
    goto step 8
else
    Stop search
end

Algorithm 1: Algorithm 1: Heuristic Algorithm

Formulation of start time, ST for computation task is as follows:

$$ST(v_{entry}) = 0 \tag{22}$$

$$ST(v_i) = \max_{v_p \in pred(v_i)} (FT(v_p)) \tag{23}$$

Formulation of finish time, FT for computation task is as follows:

$$FT(v_i) = max_{v_c \in succ(v_i)}(TT_{v_i v_c}) + ST(v_i) + CT_{ixy} \tag{24}$$

where CT_{ixy} is the computation time of v_i at svc_{xy} where $pred(v_i)$ = parent of computation task, v_i and $succ(v_i)$ = child of computation task v_i.

4 Evaluation

This section evaluates the performance of the heuristic algorithm (HA). In order to evaluate the quality of the solutions generated by the HA, we also developed and implemented another algorithm using Exhaustive Search (ES). It was guaranteed that the ES always generated an optimal solution. However, the computation of the ES increased exponentially when the problem size increased.

4.1 Experimental Environment

Both the HA and the ES were implemented using C++ programming language. All the experiments were conducted in a desktop computer with a 3.4 GHz Intel 4 Core(s) and an 8 GB RAM.

4.2 Experimental Design

The computation time and quality of the results produced by the HA depend on the size of the problem, which is dependent on two parameters. 1) The number of computational tasks in the workflow. 2) The number of available services for each of the computational task. To construct the test problem, a 10 tasks workflow was used (see Figure 2) as the building block. 10 workflows of 15 tasks were generated randomly. The values of attributes of workflow tasks and cloud services are as shown in Table 1:

Attributes	Range
Processing speed of cloud service	10,000 - 70,000 Million Instructions Per Second (MIPS)
Bandwidth between mobile device and cloud service	10 - 20 MBps
Output data size of computational task	10 - 20 MB

Table 1: Range of values of attributes

For mobile device, no price is charged, the Wi-Fi transmission power is 2.3 W and the processing speed is fixed to 2000 MIPS.

4.3 Experimental Results

We randomly generated 10 test problems, each of which had 10 tasks and each of the computation tasks had 15 computation service providers. In order to test the quality of the solutions generated by the HA, we also designed and implemented an optimal algorithm which uses exhaustive search.

The HA and optimal algorithm were used to solve a number of randomly generated test problems with 10 computational tasks. The computation times and the solutions are shown in Table 2, where CT stands for Computation Time in seconds and LFT stands for Latest Finish Time in seconds.

However, when we increase the number of services to 10 services, there are 1010 combinations which could lead to very large computation time to generate all possible combinations and find the best solution from among them. Therefore, we limit the computation time to one hour which is equal to 3600 seconds and find the best solution within that amount of computation time. The results are shown in Table 5. As can be seen from Table 5, Exhaustive Search could find similar or better solution than Heuristic Algorithm for four out of ten workflows. Meanwhile Table 6 shows that Exhaustive Search method could find better solution than HA for one workflow only out of ten workflows. This may indicate that though Exhaustive Search is capable to obtain better solution quality than our HA, but it takes very large computation time especially when the problem size increases, in this case up to 15 services. On the other hand, our HA could find good results less than one second. If the problem size is increased further, Exhaustive Search may not be able to obtain better quality solutions than our HA for all 10 randomly generated workflows within one hour.

Test Index	Heuristic Algorithm					Exhaustive Search			
	CT	Obj(X)	Price	LFT	Energy	Obj(X)	Price	LFT	Energy
1	0.049	0.971586	1.3287	10	40.5879	0.881012	1.8346	21	184.1550
2	0.042	0.963549	1.7680	8	55.4040	0.771434	0.8347	38	361.9988
3	0.063	0.980348	1.0765	8	42.0900	0.767645	0.8347	40	361.9988
4	0.050	0.975632	1.3287	8	40.9048	0.96318	0.9563	10	38.9690
5	0.053	0.977055	1.0765	9	42.0900	0.764327	0.8347	41	361.9988
6	0.056	0.977747	1.0765	7	41.8248	0.766674	0.8347	39	361.9988
7	0.048	0.973391	1.0765	7	41.5079	0.763215	0.8347	39	361.9988
8	0.063	0.970706	2.0825	7	38.5090	0.76881	0.8347	39	361.9988
9	0.069	0.967394	2.0825	11	38.5090	0.761059	0.8347	43	361.9988
10	0.058	0.974327	1.0765	9	42.0900	0.762212	0.8347	41	361.9988

Table 2: Comparison results and computation time of HA and Exhaustive Search of 10 tasks and 15 services test problem

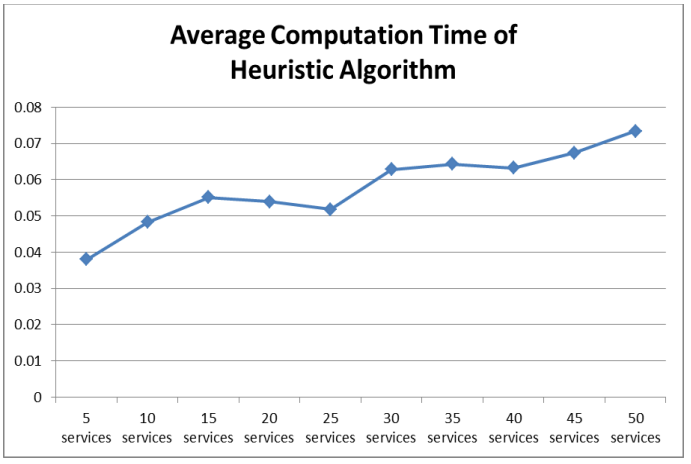

Figure 2: Effect of number of services

5 Conclusion and Future Work

This paper has formulated a new mobile computation offloading problem, namely, multi-site computation offloading in mobile cloud computing, and transformed the new computation offloading problem into a multi-objective combinatorial optimization problem. The combinatorial optimization problem has three objectives: (1) to minimize the energy consumption of the mobile device; (2) to minimize the computation time; (3) to minimize the total cost of the computation incurred by cloud computing. The multi-objective optimization problem has been transformed into a weighted single-objective optimization problem and a heuristic algorithm has been proposed for solving the weighted single-objective optimization problem. The proposed heuristic algorithm has been evaluated by experiments and the experimental results have shown that the heuristic algorithm can produce good quality solutions in a reasonable time for those test problems.

This is the first attempt on the multi-size computation offloading problem in mobile cloud computing. In the future we shall study the scalability of the heuristic algorithm. We shall also

design a genetic algorithm for the problem.

References

[1] One hour translation: translation services. [online], January 2016. https://www.onehourtranslation.com/.

[2] Luiz F Bittencourt, Edmundo RM Madeira, and Nelson LS Da Fonseca. Scheduling in hybrid clouds. *Communications Magazine, IEEE*, 50(9):42–47, 2012.

[3] Luiz Fernando Bittencourt and Edmundo Roberto Mauro Madeira. Hcoc: a cost optimization algorithm for workflow scheduling in hybrid clouds. *Journal of Internet Services and Applications*, 2(3):207–227, 2011.

[4] Byung-Gon Chun, Sunghwan Ihm, Petros Maniatis, Mayur Naik, and Ashwin Patti. Clonecloud: elastic execution between mobile device and cloud. In *Proceedings of the sixth conference on Computer systems*, pages 301–314. ACM, 2011.

[5] Eduardo Cuervo, Aruna Balasubramanian, Dae-ki Cho, Alec Wolman, Stefan Saroiu, Ranveer Chandra, and Paramvir Bahl. Maui: making smartphones last longer with code offload. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 49–62. ACM, 2010.

[6] Hoang T Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611, 2013.

[7] Yoonseok Heo, Taeseop Kim, and Doug Young Suh. Video streaming for multi-cloud game. In *Advances in Multimedia Information Processing–PCM 2015*, pages 275–284. Springer, 2015.

[8] Roelof Kemp, Nicholas Palmer, Thilo Kielmann, and Henri Bal. Cuckoo: a computation offloading framework for smartphones. In *Mobile Computing, Applications, and Services*, pages 59–79. Springer, 2010.

[9] Sokol Kosta, Andrius Aucinas, Pan Hui, Richard Mortier, and Xinwen Zhang. Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading. In *INFOCOM, 2012 Proceedings IEEE*, pages 945–953. IEEE, 2012.

[10] Karthik Kumar and Yung-Hsiang Lu. Cloud computing for mobile users: Can offloading computation save energy? *Computer*, (4):51–56, 2010.

[11] Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. Cloudcmp: comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2010.

[12] Hongzhi Li and Xian-Sheng Hua. Melog: mobile experience sharing through automatic multimedia blogging. In *Proceedings of the 2010 ACM multimedia workshop on Mobile cloud media computing*, pages 19–24. ACM, 2010.

[13] Ruben Van den Bossche, Kurt Vanmechelen, and Jan Broeckhove. Cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds. In *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, pages 320–327. IEEE, 2011.

[14] Simon S Woo and Jelena Mirkovic. Optimal application allocation on multiple public clouds. *Computer Networks*, 68:138–148, 2014.

[15] Huijun Wu and Dijiang Huang. Mosec: Mobile-cloud service composition. In *Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2015 3rd IEEE International Conference on*, pages 177–182. IEEE, 2015.

[16] Zhi Ye, Xin Chen, and Zhu Li. Video based mobile location search with large set of sift points in cloud. In *Proceedings of the 2010 ACM multimedia workshop on Mobile cloud media computing*, pages 25–30. ACM, 2010.