Big Data and Information Analytics
©American Institute of Mathematical Sciences
Volume **1**, Number **1**, January **2016**
doi:10.3934/bdia.2016.1.1
pp. **1–13**

ACO-BASED SOLUTION FOR COMPUTATION OFFLOADING IN MOBILE CLOUD COMPUTING

Weidong Bao*

College of Information System and Management
National University of Defense Technology
Changsha, Hunan, 410073, China

Haoran Ji†, Xiaomin Zhu†, Ji Wang†
Wenhua Xiao† and Jianhong Wu†,‡

† College of Information System and Management
National University of Defense Technology
Changsha, Hunan, 410073, China
‡ Department of Mathematics and Statistics, York University
Toronto, Ontario, M3J 1P3, Canada

Abstract. The cloud computing has attracted growing attentions for its benefits to providing on-demand services, mobile cloud computing (MCC) enables an increasing number of applications and computational services available on mobile devices. In MCC, computation offloading is one of the most important challenges to provide remote execution of applications to the mobile devices. Here we mainly introduce the ant colony optimization (ACO) to address this challeng and propose an ACO-based solution to the computation offloading problem. The proposed method can be well implemented in practice and presents with low computing complexity.

1. **Introduction.** The amount of data has been exploding. Trillions of bytes of data about our lives are captured to facilitate the knowledge discovery and capture the superiority of information [13]. Cloud computing is an important paradigm in IT service delivery and provides an attractive solution to many of the challenges we are facing with respect to big data. It enables a shared computing resource pool, which includes the computing resource, storage resource and information resource. These resources can be delivered as services on demand through the network. With the advances in technologies for wireless communication and the explosive growth of the mobile devices, mobile cloud computing (MCC) has been proposed as a special kind of cloud computing, which enables an increasing number of applications and services available on mobile devices [15]. Mobile users accumulate rich experience of various services from mobile applications (e.g., iPhone apps, Google apps, etc),

2010 *Mathematics Subject Classification.* Primary: 90C; Secondary: 90B18.

Key words and phrases. Mobile cloud computing, service offloading, ant colony optimization, multi-clouds, remote execution.

the National Natural Science Foundation of China under grant 61405252,61572511, the Hunan Provincial Natural Science Foundation of China under grant 2015JJ3023 as well as the Overseas, Hongkong&Macau Scholars Collaborated Research Fund of China under grant 11428101.

* Corresponding author: Haoran Ji.

which run on the devices and/or on remote servers via wireless networks. Nevertheless, many challenges remain before enjoying these convenience for the unreliable wireless network and limited bandwidth.

With the emergence of mobile cloud computing, an increasingly number of applications and services becomes available on mobile devices. Technological innovations in MCC are occurring at an accelerated rate for the reasons of the growing capabilities of mobile devices and the new developments of wireless network [19]. According to the Strategy Analytic company, there are more than ten billion smart phone users all around the world by the time of 2012 and the number will double by 2015, which has been proven to be right. Juniper research reports that the consumer and enterprise market for cloud-based mobile applications is excepted to increase by $9.5 billion up to 2014 [3]. Therefore, research interest in MCC is growing with the development of intelligent mobile terminals and the changing user habits on enjoying IT services. Compared with cloud computing normally implemented on personal computers, MCC generally presents new characteristics, some of which listed below.

- Smart Mobile Devices (SMD) are much more power than before. They are the dominant future computing devices with high user expectations for accessing computational intensive applications analogous to powerful stationary computing machines [15].
- SMDs access the network mainly with wireless method. The bandwidth is relatively limited and unreliable. The unreliability of wireless connection leads to disconnected, intermittent connectivity, or long latency. More service tactics are needed to increase the qualities of mobile cloud computing service.
- The capacities of SMDs are always inferior to the requirements of cloud-based mobile applications. People always pursue for better experience of mobile applications, which promote the development of SMD. So, remote servers should be employed in MCC to facilitate the outsourcing of intensive applications [16].

All of the above characteristics provide challenges to providing satisfied services to MCC clients. In practice, cloud servers are located in one or several places. Long distance data transmission will generate longer latency. And also longer time means risk of failure. Otherwise, the unreliable wireless will consume huge communication resource wasting on confirming the SMD states, especially for long distance data transmission. So, it is critical for MCC servers to shorten the transmission time. An effective solution is to optimize the allocation of services.

To guarantee the quality of service (QoS), mobile cloud computing environments are normally structured as decentralized cloud computing environments, in which cloud servers located in distributed places [14]. These servers are normally with different capacities and properties. So, scheduling the services among these distributed servers is an effective solution to optimize the services of MCC. This line of research addresses the scheduling problem of allocating the services on distributed servers. This approach has also been proposed to be employed in the computational offloading frameworks, as these involve an offloading problem in MCC.

This paper addresses the issues of optimizing the service provision to SMD in MCC environments. We formulate this as a multi-cloud problem, and each cloud presents heterogeneous properties of cost, service time and so on. This introduces the hybrid clouds to the computation offloading problem and allow us to focus on the issues of service allocation among distributed servers. Our work focuses on the bandwidth, which always presents upper limit during remote execution of mobile applications, and well balance these two challenges. To our best knowledge, little

work has been done in the literature about this special key researching problem on MCC. We outline the contributions as follows:

- We give a detailed description on the problem of service allocation in MCC, and we formulate this problem to clarify the process of optimization.
- We employ the Ant Colony Optimization (ACO) method to optimize the scheduling. Our proposed solution is feasible in practice and can well solve the problem with low complexity.
- We propose a solution for implementing the proposed algorithm, and we evaluate the performance.

The remaining part of the paper is organized as follows. Section 2 gives a full introduction on the related work. Section 3 describes the motivation of this paper and gives a formulation of the problem. In Section 4, a hybrid intelligent optimization algorithm is proposed and the detailed solution is presented with an illustrative case. Finally, Section 5 concludes the paper and shows the perspectives of this work.

2. **Related work.** A lot of studies have been done to promote the development of MCC. A well-accepted definition of MCC is "Mobile Cloud Computing at its simplest, refers to an infrastructure where both the data storage and the data processing happen outside of the mobile device. Mobile cloud applications move the computing power and data storage away from mobile phones and into the cloud, bringing applications and mobile computing to not just smart phone users but a much broader range of mobile subscribers" [1]. AEPONA[2] describes MCC as a new paradigm for mobile applications whereby the data processing and storage are moved from the mobile device to powerful and centralized computing platforms located in clouds. MCC bridges the disparity between the computing resources of SMDs and processing requirements of intensive applications on SMDs.

Recently, a number of computational offloading algorithms have been proposed for intensive applications to remote servers. The best examples of distributed models for offloading algorithms include: the decentralized virtual cloud computing environment for mobile devices, the centralized cloud computing environment for mobile devices, and the centralized cloud computing data centers based cloud computing environment [15]. These centralized applications are then accessed over the wireless connection based on a thin native client or web browser on the mobile devices. Remote execution of the services on SMDs, well-known as computation offloading, is one of the most attractive problems in MCC. Computation offloading is to send heavy computation to resourceful servers and receiving the results from these servers. For these issues, Shiraz et al. in [14] propose an application offloading framework with dynamic partitioning mechanism. Dynamic partitioning techniques continuously or periodically evaluate the availability of resources on mobile device and the requirements of the mobile application. The proposed strategy has been testified to be effective and stable under MCC circumstances by practical experiments on several development boards. Abolfazli et al. [4] introduce the hybrid cloud mode into MCC and gives a description on the motivation, taxonomies and challenges on the issues. Bittencourt L.F. et al. [6] follow the work in [7] and concern the scheduling problem of workflows in hybrid clouds, in which tasks can be allocated either on private clouds or on public clouds, and emphasize the communication capacity with a prominent importance. Generally speaking, the MCC is an emerging technology and is still waiting for further research.

As the limitation of resources is one of the key characteristics on MCC, optimizing the utility of network resource is a major objective of this research [20]. Therefore, load balancing is the major objective of this paper, as it is one of the most efficient solutions for the problems of limited resources. A common practice of load balancing in data centers is deploying front-end load balancers to direct each client's request to a particular server replica to increase throughput, reduce response time, and avoid overloading of network. Wang et al. in [18] make initial efforts to build a model and develop algorithms for load balancing. The algorithms compute concise wildcard rules that achieve a target distribution of the traffic, and automatically adjust to changes in load-balancing policies without disrupting existing connections. Koerner et al. in [11] introduce differentiated load balancing algorithms for different types of traffic, for example, web traffic and email traffic, to achieve dedicated and specialized balancing algorithm implementations depending on requirements of services and workloads. To solve this problem the load balancer becomes the bottleneck of a data center, Handigol et al. in [10] present Plug-n-Serve, which balances load over an arbitrary unstructured network and directly controls paths taken by sending new requests to minimize average response time of web services.

Although much effort has been made to promote the development of MCC. Progresses on the hybrid clouds in MCC is lacking. To the best of our knowledge, little has been done on the issues of service allocation among multiply cloud servers in MCC.

3. Motivation and formulation.

Normally, there are three main components of MCC, which are the client terminals, the cloud servers and the delivery network. The client terminals, which in most cases are the SMDs, request for services of computation. And the cloud servers typically consists of several clusters of servers or personal computers to storage and provide services to the clients. The clusters tend to be distributed in different places. The delivery network connects the clients and cloud servers, of which wireless connections present as key components.

In MCC environments, client terminals get services including applications, computing and private data from cloud servers through the delivery network. Nevertheless, in practice, it is nothing out of ordinary for an application or operating data to consume certain size of the bandwidth during the transmission. So, the choice of service allocation should take into account the network environments, the servers, and the task requirements.

The framework of MCC can be generally described as Fig. 1. We formulate the problem generally by the following aspects: the clients, the cloud servers, and the delivery network.

3.1. Clients.

The clients, denoted as C_i, are the terminal devices that access to the network. They are normally the SMDs. Terminal devices can be generally separated into two kinds: mobile clients (MC_i) and PCs (PC_i). And this paper just focus on the mobile terminal devices. Although the abilities of mobile devices have increased a lot, it is still limited and not powerful to support complex interactive operations, such as drawing HD images, running high performance games and so on. The services that C_i request for, denoted as S_i, can be split into three parts, separately is services of computing (SC_i), services of data storage (SD_i) and services of information (SI_i). And computation offloading mainly concerns the provision of SC.

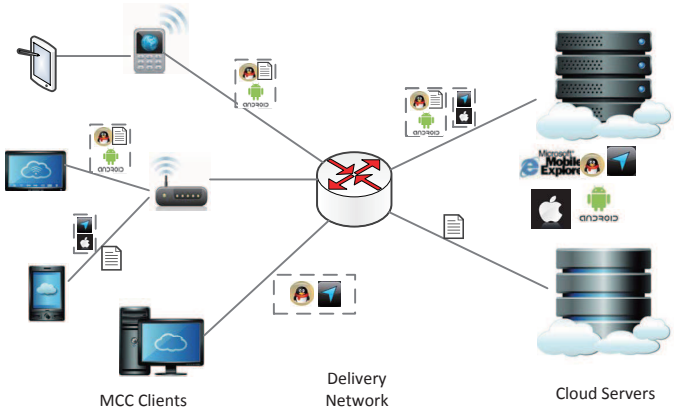

FIGURE 1. Description of Mobile Cloud Computing

A terminal device C_i has a property of the maximum transmitting speed, denoted as $maxS$, which describe its ability on transmitting data through the network. The requests for SC_j should normally contain several aspects of the workload of service, the time limitation, data transmission size and memory requirement. Workload of service, denoted as $WLoad_j$, is a description on the load requests on computational resources. Usually, it can be measured as the millions of instructions per second ($MIPS$). Time limitation, denoted as TL_j, is a description on the time constraint. The remote execution of computation should be completed within a certain time limitation. Exceeding the limitation will reduce the quality of service, for instance the "1s principle" in IT. Data transmission size, denoted as $DLoad_j$, represents the Memory requirement, denoted as Mt_J, describes a threshold of SC_j for memory. For different applications, there can also be other threshold requirements for other capacities. It is a description the qualification to be an offloading server.

3.2. **Cloud servers.** The cloud servers is normally a powerful storage center in the MCC framework. It can be treated as a data center, denoted as DC, similar to cloud computing. It typically consist of a cluster of servers or personal computers or both. A server, denoted as Sr_i, may provide certain types of services. Servers provide services to the clients when they receive service requests. Accordingly, servers mainly provide the following three kinds of service as the computing service, the data storage service and the information service. An Sr_i will present a certain capacity on computing. It will provide certain types of computing services. The computing capacity is described as Cr_i. Usually, schedulers are used to manage the resources in a server or in a DC.

For the network environments of these servers are normally with good conditions and wired solution is the key characteristic of data transmission, loads on the bandwidth of servers are more likely to be low in MCC. The property of bandwidth (Bw_{Sr}) is used to measure this capacity. These servers may be located in different places and each location has its properties. So these servers present with different network environments, servicing costs, energy consumptions and so on. Allocating the services should take consideration of these properties.

3.3. **Delivery network.** Delivery network is the connection between clients and cloud servers, which can be represented as a graph $G = (V, E)$, in which $v \in V$

represents an entity in the network. $e_{ij} \in E$ represents a relationship with property e between entities v_i and v_j. A v_i can be a client, a router or a server. So, for a network, V can be described as a sequence of devices as:

$$V = \{C_1, C_2 \cdots C_i, R_{i+1} \cdots R_j, Sr_{j+1} \cdots Sr_k\} \tag{1}$$

Where $k > j > i$.

The delivery network has much to do with the service time. The network environments consist of many types of network equipments. For the bandwidth of each line and the capacities of routers are fixed, services face up with different network environments. And the available bandwidth may vary under the TCP/IP agreement. If a line is busy, the available bandwidth will accordingly smaller. So the allocation of services will vary with the network environments. Big data leads to heavy traffic load in the network, which is especially significant in MCC.

To address these challenges, many efforts have been made. The new frontier developed recently of software defined network (SDN) attracts lots of interests and make it possible to design the path of data transmission, which can improves the network performance. The core idea of SDN can be generally described as in Fig. 2. In SDN, a controller manage several switching devices, such as routers. And the data transmission can be managed and optimized by the controllers. However, SDN do not present good performance on large-scale network. The communications and synchronization among different controllers restrict the performance. It is impossible to construct a controller with so powerful to manage the whole network. Similarly, it is challenging to construct a scheduler with the ability to realize global optimization.

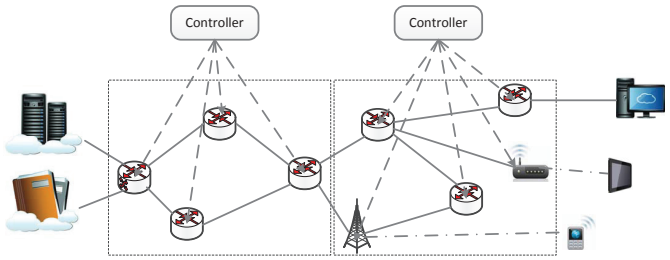

FIGURE 2. The SDN solution for service offloading in MCC

A link, denoted as L_{ij}, describes the date transmission path from Sr_i to C_j. There exists a bandwidth for each L_{ij}, denoted as Bw_{ij}, which describe the smallest bandwidth in the path. And also a parameter bw_{ij} is used to describe the minimum bandwidth of the line between v_i and v_j. A line between two network devices including the routers and terminals, denoted as l_{ij}, has a property of workload $load_{ij}$. $load_{ij}$ is determined by the amount of transmission date in the path. Although a pre-set bandwidth of the network is easily to get, which describe the ideal status. However, in practice the bandwidth, especially for wireless delivery network, in most cases is less than the pre-set value. bw for MCC presents unstable and fluctuant, sometimes even be zero.

In MCC, it is impractical to employ a global scheduling method. Researches on optimizing the service offloading in MCC should take full consideration of the practicality of proposed solution.

3.4. **Service time.** As mentioned in the introduction, service time is an important measure of the QoS. Service time mainly consists of the computing time and the transmission time. The transmission time, denoted as Ts^k, of service S_i can be computed as equation 2. Ts^k_{ji} is decided by the data size of S_k and the bandwidth of the link targeting client C_i.

$$Ts^k_{ji} = \frac{DLoad_k}{Bw_{ji}} + Latency_{ji} \tag{2}$$

Where $Latency_{ji}$ is the time delay of transmission from v_j to v_i.

The computing time of service S_i, denoted as Tc_k, can be computed as equation 3.

$$Tc^k_{ji} = \frac{WLoad_k}{Cr_j} + \gamma \tag{3}$$

Where γ is a slack variable to formulate the variability.

The scheme of service providing is defined as $C_i = Scheme(Sr_j, S_k, L_{ji})$, which denotes that service S_k is provided from Sr_j to C_i through L_{ji}. And there is a threshold of TL^t_k for each S_k, which describes the time constraint for mobile cloud services. So, the service time can be described as $Ts^k_{ji} + Tc^k_{ji}$ [12]. If $Ts^k_{ji} + Tc^k_{ji} > TL^t_k$, it means that C_i enjoys S_k with a low QoS. If $Ts^k_{ji} + Tc^k_{ji} \leq T^t_k$, it means that C_i get S_l with a satisfying QoS. The ratio of satisfying services is an important measurement to evaluate the computational offloading mechanism in MCC.

Based on the problem formulation, allocating the services to servers should take consideration of the requests of clients, the delivery network and the states of servers. And also, it is unpractical to employ a strong controller of global network as a solution. So, this work aims at exploring a practical solution of implementing the service offloading in MCC.

4. **Service offloading in MCC.** Offloading becomes an attractive solution for meeting response time requirements on mobile systems as applications become increasingly complex [5]. And the offloading can be generally described as Fig. 3.

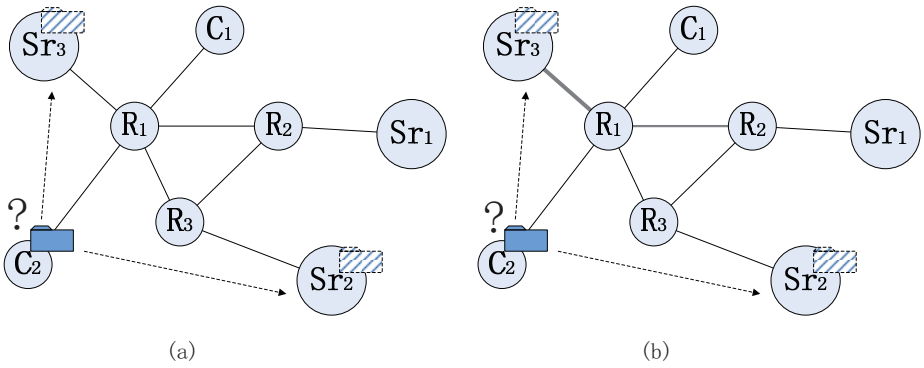

FIGURE 3. Description of service offloading in MCC

In MCC environments, it is important to provide timely services, which asks for sufficient bw. Assuming that a client C_1 requests for the service S_1 and that

the network structure is constructed as Fig. 3. In Fig. 3 (a), it detects that Sr_2 and Sr_3 can provide satisfying remote execution services. Then it should decide from which client to get the service is better. But, it hesitates on which one C_1 should get the offloading service. As the aims are to provide real-time service, the server with earlier service time will be chosen as the offloading object. So, the decision concerns on the service time of each choice. On this occasion, the offloading scheduling algorithm should be triggered in order to ensure the QoS. Assume that the decision is $C_2 = Scheme(Sr_3, S_1, L(Sr_3, C_2))$. In Fig. 3 (b), it is congestion of $l(R_1, Sr_3)$. So, the data transmission time from C_2 to Sr_3 will be very big, which can not be accepted by the clients. On this condition, the scheme should be changed as $C_2 = Scheme(Sr_2, S_1, L(Sr_2, C_2))$. Meanwhile, other requests from C_1 and so on also do their offloading in Sr_2. Therefore, 1 second later, $l(R_1, Sr_3)$ may be unblocked and $l(R_3, Sr_2)$ would be blocked. So, realizing the global optimization is effective on improving the offloading performance in MCC.

In MCC, employing the method of monitoring the network is an available solution. But, it will cause huge burden, which presents inefficient, and the allocation of the scheduler is also a challenge. So, a practical solution is deeply needed for these issues.

4.1. **Ant colony optimization.** Ant colony optimization (ACO) initially proposed by Marco Dorigo in 1992 in his PhD thesis [8][9] aims to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. It has been applied to many combinatorial optimization problems, ranging from quadratic assignment to protein folding or routing vehicles and a lot of derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations[17]. In ACO, there are m independent ants and they obtain m different solutions or paths by choosing the next node to be appended to their respective tour iteratively. When an ant reaches a node and chooses the next ongoing edge, it parses its options. It categorizes the options into sets of nodes: the ones which are promising because no individual of its rule-instantiation has walked them earlier and the ones that already have been visited. If promising options are available, one of them is chosen by random. Otherwise the ongoing path is chosen probabilistically by preferring less visited triples. The general ACO algorithm can be summarized in a pseudo code shown in Algorithm 1.

ACO is a typical multi-agent application with positive feedback. It can well fit the searching problem related with path programming. Each client, which sends the request, can be treated as the colony and the services located on available servers are treated as the food. Ants start their searching from the colony. Assume that ants will not go back to the entities it has visited. So, the offloading problem can be translated as searching for the food for ants in that colony. To employ the ACO method, some basic definitions should be given.

Definition 4.1. The distance between v_i and v_j, denoted as D_{ij}, in G is defined as the transmission time of 1M data from v_i to v_j.

With equation 2, the definition of D_{ij} can be computed by equation 4.

$$D_{ij} = \frac{1}{Bw_{ij}} + Latency_{ij} \tag{4}$$

In the equation, D_{ij} is determined by Bw_{ij} and $Latency_{ij}$.

Algorithm 1: ACO

```
Initialize the base attractiveness, τ, and visibility, η, for each edge;
for i < IterationMax do
    for each ant do
        choose probabilistically to move into the next state;
        add that move to the tabu list for each ant;
        repeat until each ant completed a solution;
    end
    for each ant that completed a solution do
        update attractiveness for each edge that the ant traversed;
        if local best solution better than global solution then
            save local best solution as global solution;
        end
    end
end
```

Definition 4.2. Selecting probability is the transition probability of an ant from one state to another state, denoted as P^k.

The probabilistic section rule apply searching by selecting node randomly according to a probability function shown in equation 5.

$$P_{ij}^k = \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_{u \in U_i^k} (\tau_{ij})^\alpha (\eta_{ij})^\beta} \tag{5}$$

Where P_{ij}^k is probability for the kth ant to choose the next entity v_j from v_i using pheromone, η_{ij} is the problem-specific heuristic information that represents a priori information about the problem instance definition, or run-disinformation provided by a different source other than ants, τ_{ij} is the pheromone for the link between v_i and v_j, U_i^k the set of unvisited nodes for the ant^k and α and β are parameters controlling the relative influence of the pheromone and local heuristic values respectively in the probabilistic selection of the next entity to visit. If $\alpha = 0$ the algorithm behaves as a standard greedy algorithm, with no influence from pheromone. If $\beta = 0$ only pheromone amplification will occur. The selection for the next node is a probabilistic one. An edge with a large local heuristic values called semantic score increases the probability of that edge being chosen.

Definition 4.3. Pheromone is a measure of the number of ants who have creep through the path, denoted as τ.

When an ant arrives at a place, it will deposited pheromone in the place to mark its searching path. After it has completed a solution, the trails of τ will be updated by equation 6.

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_k \Delta\tau_{ij}^k \tag{6}$$

Where τ_{ij} is the amount of pheromone deposited in e_{ij}, ρ is the pheromone evaporation coefficient and $\Delta\tau_{ij}^k$ is the amount of pheromone deposited by kth ant. After an ant reaching a food, it will report to the scheduler and give a candidate solution. And then the pheromone will be renewed.

With the ACO method, ants will just communicate with the scheduler once, which happens when they come to an available solution. As the problem is a real-time scheduling problem, the scheduler should get the solution within a certain time, denoted as T^s. And there is the constraint of $T^s + T \leq TL^t$, where $T = Ts + Tc$. Assume that a colony will sent m ants each turn. And ants leave with the interval of t'. So the number of turns, denoted as N , should satisfy:

$$N \leq \frac{T^s}{t'} = \frac{TL^t - T}{t'} \tag{7}$$

4.2. **ACO based solution.** Global optimization requirements are key challenges for the scheduler. To deal with these issues, this paper proposed an ACO-based method. On the condition of many SMDs requesting for services at the same time or nearly the same time, the method treats each SMD as a colony. To optimize the overall performance, it introduces the following rules:

Rule Ants in one colony only search for one special kind of food.
Rule Ants prefer to choose the path with more pheromone of other ants coming from the same colony.
Rule Ants prefer to choose the path with less pheromone of other ants coming from different colonies.
Rule Ants prefer to choose the path with wider bandwidth, which normally means they can creep faster.
Rule The pheromone will decrease by the time but as least last for T^s.
Rule The pheromone will be nearly zero at the time of T^s.

With these rules, original ACO parameters will be set as follows:

$$\rho = \frac{1}{T^s} \tag{8}$$

$$\eta_{ij} = \frac{bw_{ij}}{\xi_{ij}} \tag{9}$$

Where ξ_{ij} is the pheromone of the ants from other colonies for the link between v_i and v_j and bw_{ij} is the smallest bandwidth in the link from colony to its position.

$$\Delta\tau_{ij}^k = \frac{\kappa D_{ij}}{LL^k} \tag{10}$$

Where LL^k is the path length of the solution the kth ant has found and κ is a constant. So equation 5 and 6 can be converted to:

$$P_{ij}^k = \frac{(\tau_{ij})^\alpha (\frac{bw_{ij}}{\xi_{ij}})^\beta}{\sum_{u \in U_i^k} (\tau_{ij})^\alpha (\frac{bw_{ij}}{\xi_{ij}})^\beta} \tag{11}$$

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_k \frac{\kappa D_{ij}}{LL^k} \tag{12}$$

With above work, Pseudo code of the ACO-based solution can be described as in algorithm 2. In algorithm 2, the initial time of T_0 is set as zero and the real time is denoted as T, W^i is the position of jth ant, W^s is the position with requesting service, $Path^i$ is the path of ith ant has visited.

When a client sends service requests for remote execution, it will run the proposed algorithm to generate the solution. In the proposed method, the decision process can be presented by the path choice. And algorithm 2 is a description of the service offloading process.

Algorithm 2: ACO-based service offloading solution

Load $G = (V, E)$;
Initial T_0, T, m, t', N;
for *each* e_{ij} **do**
    Compute $D_{ij} = \frac{1}{Bw_{ij}} + Latency_{ij}$;
end
for *i=1:N* **do**
    for *each* e_{ij} **do**
        Get pheromone information of G from the scheduler;
        Compute newest τ_{ij};
    end
    for *j=1:m* **do**
        Initial the state of jth ant in ith wave;
        Find available v for the next step;
        if *No available* v **then**
            Ant die;
            Return;
        end
        while W^j *!=*W^s *and Number of available* $v > 1$ **do**
            Generate $P_{ij}^k = \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_{u \in U_i^k} (\tau_{ij})^\alpha (\eta_{ij})^\beta}$;
            Select node according to P_{ij}^k;
            Record the $Path^j$;
            Renew the W^j;
            Deposit pheromone on the path;
        end
        if W^j *==*W^s **then**
            Get LL^k;
            Renew $\tau_{ij} = (1-\rho)\tau_{ij} + \sum_k \frac{\kappa D_{ij}}{LL^k}$;
            Report to the scheduler;
        end
    end
    Predict T based on choosing preference;
    Renew N;
end

4.3. **Implementation of the ACO-based solution.** The implementation of computation offloading in MCC brings new challenges. To further describe the implementation of the proposed ACO-based solution, an application case is described as in Fig. 4.

In Fig. 4, the boldness of line indicates the traffic load and the color indicates the pheromone strength of ants from other colonies. If there are clients requesting for services, they will repeatedly sent messages with size of about 1 KB or even smaller. Each message can be treated as an "ant". The information carried by the "ant" includes the entities it has visited and the pheromone it has deposited. For instance, C_1 request for the computational service on remote servers. So it send its "ants" to search for the service. When the "ant" arrives at a router, e.g. R_5, it will make the selection process according to P_{ij}^k as shown in algorithm 2. Once

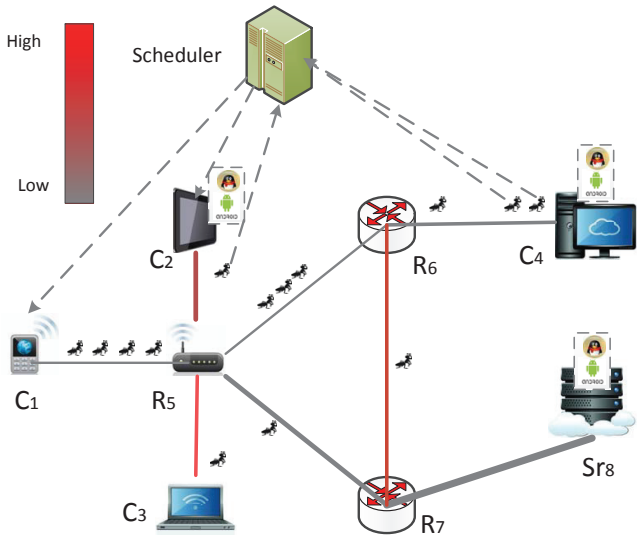

FIGURE 4. Case Study

it arrives at the device, which has the service it searching for, it will report and transmit its searching information to the scheduler. After receiving the report, the scheduler will renew the total knowledge about the pheromone distribution in the network and deliver the knowledge to each device in MCC including the routers. The schedulers will renew the P_{ij}^k for the coming "ants" based on these knowledge and send these information to the clients. At last, the scheduler will give a decision before T^s for each request.

In above case, the ACO-based method do as an effective solution for the problems of computation offloading in MCC.

5. **Conclusion.** MCC has been a novel pervasive computing mode, which can be treated as a paradigm of cloud computing. With the rapid development of mobile technology, MCC has been attracting more and more attentions. Meanwhile, the computation offloading problem has been a challenge in MCC. This paper aims to address the issue of the computation offloading in MCC and proposes solutions for challenges which hinder the implementation of MCC. In particular, we address the issues of allocating the services to the distributed servers. To solve the problem, an ACO based solution is proposed. This solution develops the ACO with a multi-agent mode and makes a global optimization on the service providing for mass clients. The proposed method defines different colonies and ants coming from one colony will prefer to eat a certain kind of food. Also some other rules, for example the ants will exclude the path with more pheromone deposited by the ants from other colonies, are introduced to adapt for large number of service requests. And finally, a case is used to give a further understanding on the implementation of the proposed method.

REFERENCES

[1] 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, *Wireless Communications and Mobile Computing*, **13** (2013), 1587–1611.

[2] White Paper, *Mobile Cloud Computing Solution Brief*, AEPONA, November 2010.
[3] R. Holman, *Mobile Cloud Computing: $9.5 billion by 2014*, http://www.juniperresearch.com/analyst-xpress-blog/2010/01/26/mobile-cloud-application-revenues-to-hit-95-billion-by-2014-driven-by-converged-mobile-services/, (2010).
[4] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani and R. Buyya, Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open issues, *IEEE Communications Surveys and Tutorials 2014*, **16** (2014), 337–368.
[5] R. K. Balan, *Simplifying Cyber Foraging*, Ph.D thesis, School of Computer Science, Carnegie Mellon University, 2006.
[6] L. F. Bittencourt, E. R. M. Madeira and N. L. S. D. Fonseca, Scheduling in hybrid clouds, *IEEE Communications Magazine*, **50** (2012), 42–47.
[7] L. F. Bittencourt, HCOC: A cost optimization algorithm for workflow scheduling in hybrid clouds, *Journal of Internet Services & Applications*, **2** (2011), 207–227.
[8] A. Colorni, M. Dorigo and V. Maniezzo, Distributed optimization by ant colonies, *actes dela premiare conference europeenne sur la vie artificielle*, (1991), 134–142.
[9] M. Dorigo, *Optimization, Learning and Natural Algorithms*, Ph.D thesis, Politecnico di Milano, Italy, 1992.
[10] N. Handigol, S. Seetharaman, M. Flajslik, N. McKeown and R. Johari, *Plug-n-Serve: Load-balancing Web Traffic Using OpenFlow*, ACM SIGCOMM Demo, 2009.
[11] M. Koerner and O. Kao, Multiple service load-balancing with Open-Flow, in *2012 IEEE 13th International Conference on High Performance Switching and Routing (HPSR)*, (2012), 211–214.
[12] K. Kumar, J. Liu and Y. H. Lu, A survey of computation offloading for mobile systems, *Mobile Networks and Applications*, **18** (2013), 129–140.
[13] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh and A. H. Byers, Big data: The next frontier for innovation, competition, and productivity, Online Report, 2011.
[14] M. Shiraz, E. Ahmed, A. Gani and Q. Han, Investigation on runtime partitioning of elastic mobile applications for mobile cloud computing, *Journal of Supercomputing*, **67** (2014), 84–103.
[15] M. Shiraz, A. Gani, R. H. Khokhar and R. Buyya, A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing, *IEEE Communications Survers & Tutorials*, **15** (2011), 1294–1313.
[16] M. Shiraz, M. Sookhak, A. Gani and S. A. Ali Shah, A study on the critical analysis of computational offloading frameworks for mobile cloud computing, *Journal of Network and Computer Applications*, **47** (2015), 47–60.
[17] V. Viswanathan and I. Krishnamurthi, Finding relevant semantic association paths using semantic ant colony optimization algorithm, Soft Computing, Punlished online, 22 Feb., 2014.
[18] R. Wang, D. Butnariu and J. Rexford, OpenFlow-based server load balancing gone wild, in *Proceedings of the 11th USENIX conference on Hot topics in management of internet, cloud, and enterprise networks and services (Hot_ICE'11)*, (2011).
[19] X. Zhu, C. Chen, L. T. Yang and Y. Xiang, ANGEL: Agent-based scheduling for real-time tasks in virtualized clouds, *IEEE Transactions on Computers*, **pp** (2015), p1.
[20] X. Zhu, R. Ge, J. Sun and C. He, 3E: Energy-efficient elastic scheduling for independent tasks in heterogeneous computing system, *Journal of Systems and Software*, **86** (2013), 302–314.

Received July 2015; revised August 2015.

E-mail address: wdbao@nudt.edu.cn
E-mail address: hrji@nudt.edu.cn
E-mail address: xmzhu@nudt.edu.cn
E-mail address: wangji@nudt.edu.cn
E-mail address: wenhuaxiao@nudt.edu.cn
E-mail address: wujh@mathstat.yorku.ca