Availability-aware Mobile Service Composition Over Oppotunistic Networks

XX X, XX X, *Senior Member, IEEE*, XX X, *Fellow, IEEE,*

Abstract—An opportunistic link between two mobile devices or nodes takes place when they are within communication range of each other. Typically, cyber-physical environments comprise a number of mobile devices that are potentially able to establish opportunistic contacts and serve mobile applications in a cost-effective way. Opportunistic mobile service computing is a promising paradigm capable of utilizing the pervasive mobile computational resources around users. Mobile users are thus allowed to exploit nearby mobile services to boost their computing power without investment into their own resource pool. Nevertheless, various challenges, especially its quality-of-service (QoS) and optimal scheduling, are yet to be addressed. Existing studies and related scheduling strategies consider mobile users to be fully stable and available. In this paper, instead, we propose a framework named mobile service opportunistic network (MSON) and an availability-aware QoS model for service composition. We then formulate the problem into an optimization problem and utilize an improved Krill-Herd algorithm to solve it. Finally, we carry out a case study based on some well-known scientific workflows and a real-world dataset (the D2D contact traces of MIT Reality dataset and the QoS data of QWS dataset). The comparison implies that our proposed approach outperforms traditional approaches, especially those considering stable and fully available mobille services.

Index Terms—Mobile Computing, Mobile opportunistic network, Mobile Service Composition, Service-Oriented Architecture, Service availability.

---◆---

1 INTRODUCTION

RECENT YEARS have witnessed the rapid development of mobile devices (e.g., smartphones, tablets, wearable devices, etc.) and mobile communication. Mobile devices are changing the way people getting the information and the peoples daily lives because they allow you multiple ways of communicating almost anywhere at anytime [1].

The number of mobile devices is still booming and it has already surpassed stationary Internet hosts. Mobile services are also developed and provided at a significant rate, at the same time, the requirements from mobile users are becoming more demanding, i.e., more complicated applications are needed to be run on mobile devices such as virtual reality applications [2] or machine learning applications on mobile phones [3] on mobile phones. However, because of the limited hardware resources of mobile devices (e.g., computational resource, battery life, memory, and storage), these resources-intensive tasks are usually offloaded to mobile computing cloud [4], which result in high data transfer costs (energy cost and communication fee) and high latency.

Opportunistic computing is promising complementary to conventional mobile cloud computing. As illustrated in Fig.1, the basic idea of opportunistic computing is to allow the users to utilize the resources and services that other users share, by exploiting the direct physical contacts between the users, and the resulting potential to exchange data through a direct connection between their devices (e.g. through wireless connection or Bluetooth). Resources and services available on mobile devices can be directly shared among users in a elastic and on-demand way without time-consuming and energy-requiring interactions with pre-existing infrastructure, either at the networking level (e.g., cellular networks) or at the computing/service level (e.g., the cloud). Note that, mobile tasks usually require huge computational resources or data transfer (e.g., Tensorflow
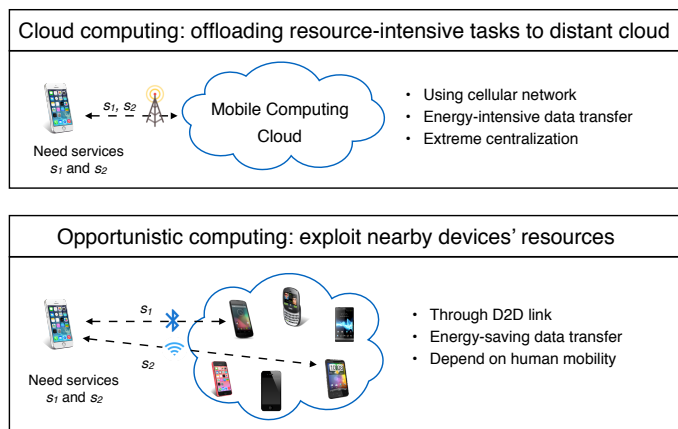

Fig. 1. Opportunistic computing

on mobile, Video editor on mobile, Online video). Nearby mobile service provider are thus more adept, in terms of energy-efficiency, at executing these tasks than the online services or nodes with the help of device to device (D2D) communications such as Bluetooth, WiFi and NFC [5]. D2D communications are featured by extensively-reduced data transfer delays and required energy than traditional cellular network. Thus it provides better user-perceived service quality in terms of reduced waiting time and improved service responsiveness. It is promising to replenish traditional cellular communications in terms of user throughput increase, cellular traffic reduction and network coverage extension.

However, due to the completely different application patterns compared with traditional service computing, service computing in mobile environment faces two inherent

challenges.

1) Constant Mobility: Mobile users may change their locations vary frequently in mobile environment, which results in the variation of service availability. Thus, determining how to handle user mobility is a major challenge for providing reliable mobile services in highly dynamic mobile environments.

2) Limited Resource: Mobile devices have limited computing capability compared with other stationary hosts. And mobile service composition plan must be generated as fast as possible because the availability of mobile service may vary much within a short time. Therefore, how to design an appropriate service selection and composition algorithms which have fast convergence rate and good scalability is another key challenge.

To address the aforementioned challenges and concerns, we propose an availability-aware mobile service composition approach in this paper, where a mobile user in mobile service opportunistic network are allowed to combine and exploit, through D2D communications, nearby devices' resources with time-varying availability (in contrast, traditional approaches mainly consider fully-available mobile services) to boost their computing power and therefore overcome the limitations of their own resources [6].

The main contributions are:

1) We propose a framework (mobile service opportunistic network, MSON in short) to address the problem of service provisioning in the mobile encounter environment where both service requesters and providers are nonstationary and with time-varying availability. In such environment, mobile user can invoke services exposed by nearby mobile devices through D2D links.

2) For MSON, we propose a availability-aware QoS model for service provisioning to capture users' mobility behavior.

3) Based on MSON and the proposed mobile service QoS model, we formulate the mobile service composition problem to an optimization problem and propose a Krill-Herd based algorithm to solve it.

2 RELATED WORK

2.1 mobile service composition

Mobile service composition refers to the technique of creating composite services with the help of smaller, simpler and easily executable services or components over mobile networks. Recent technological advances in novel mobile device design and development as well as wireless networking materialize a vision where devices all around a user, either embedded as a part of smart spaces, or being carried by other users near by, are enabled to present services probably useful. Users sometimes look for services that are not pre-existent on any device but can be dynamically built by appropriately combining already existing ones. For this purpose, extensive research efforts are carried out in this direction. For example, Deng et al. [7] classify mobile service composition methods into three categories: C2M, M2M, Hybrid. They also discussed the challenge toward mobile service provisioning and mobile service composition in terms of performance, energy and security perspective. They also proposed a mobile-service-sharing-community

model and extend the random way point (RWP) model to capture user mobility. They utilize the meta-heuristic algorithm to decide the optimal compositional plan. Yang et al. [8] present a comprehensive QoS model specifically for pervasive services. They consider not only mobile wireless network characteristics but also user-perceived factors. They derive a corresponding formula to calculate the QoS criterion. zhang et al. [9] consider a context-aware service selection algorithm based on Genetic Algorithm, They introduce a tree-encoding method to improve the capacity and efficiency of GA. However, for simplicity of the proposed model, they do not consider user mobility. Wang et al. [10] model dependable service composition in wireless mobile ad hoc networks by considering mobility prediction of the service providers. They use a probability-free model and a probabilistic model to characterize the uncertainty of composing. A service that can tolerate a certain level of the mobility of service providers. However, for simplicity of their proposed model, they focus on only the sequential workflows.

2.2 service composition over opportunistic network

Opportunistic networking is one of the most promising evolutions of the traditional multi-hop networking. Instead of relying itself on stable end-to-end paths as in the Internet, opportunistic networks do not consider node mobility a problem but as an useful opportunity. Marco et al. [11] give a review of opportunistic network and regarded it as the first step in people-centric networking, they also discuss the focused research problem such as mobility model and routing problem. Turkes et al. [12] proposed a middleware named Cocoon to support mobile opportunistic network, they design a routing protocol above Wi-Fi and Bluetooth standards, their experiments which use real-world data setups show that Cocoon performs well on the aspects of dissemination rate, delivery latency and energy consumption. Fortuna et al. [13] presented an review of dynamic service composition over both wired and wireless environment, However, their work does not present any technical details to describe how to composite service in mobile networks. Giordano et al. [6] proposed a novel paradigm that utilize Opportunistic computing as an appealing complement to the mobile computing cloud, in this way, mobile device can combine and exploit heterogeneous resources from other devices. Pu et al. [14] presented QoS-oriented self-organized mobile crowdsourcing framework, in this work, the prevalent and sufficient characteristics of opportunistic user encounters in our daily life are utilized to solve crowdsourcing problem.

3 MOBILE SERVICE OPPORTUNISTIC NETWORK

In this section, we first introduce the characteristics of mobile service opportunistic network (MSON), then a user case is presented to illustrate its application scenario, it is also assumed that MSON has the following properties:

1) Locality: Rather than stable internet, an MSON bases itself on mobile networks and exploits user mobility. Mobile users in MSON can perceive nearby services and establish self-organized local communication within permitted transmission distance.
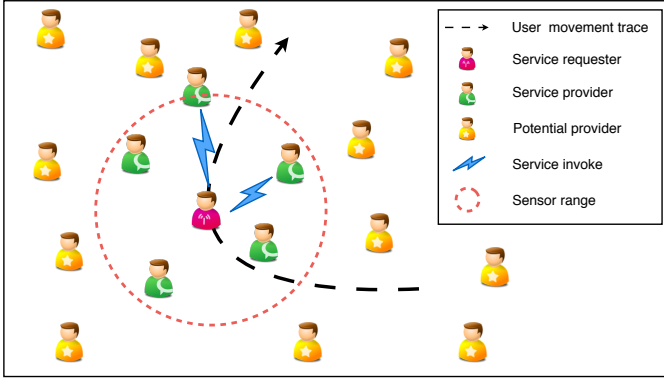
Fig. 2. Mobile service opportunistic network

Fig. 3. Mobile service availability

2) Mobility: Service requesters and providers keep moving in the mobile network even when they are invoking or provisioning mobile services.

3) Nondeterminism: Mobile services shared in an MSON are transient because the relative distance between any two services keeps changing and could rise above the permitted transmission distance at any time.

Fig. 2 illustrates how the mobile services provision over MSON. In an MSON, a mobile service requester can perceive mobile services exposed by nearby devices through D2D links and launch a request for mobile service composition. A composer process, which is in charge of discovering available mobile services nearby, selecting appropriate concrete services, and composing selected services. All concrete services interact with the composer directly [15].

Note that, we consider only one-hop D2D links for both service requesters and providers. Because D2D communications which hops are larger than two would incur network overhead [16] while one-hope communications can lower the delay (e.g., no need to transfer a large volume of task contents hop by hop) and ensure framework choose only local relatively reliable service. Besides, some existing researches [17], [18], [19], [20], [21] reveal that users' one-hop neighbors are sufficient enough, compared with multi-hop mechanisms.

We use an simple user case to illustrate the related features of service provision over MSON. Consider a mobile user who is in a crowded subway and his mobile phone has low battery. Now he wants to edit some videos, add some effects and share these video clips to his friends. If he do all these operations on his own mobile phone, his mobile phone will run out of energy because of limited battery. As one option, he can upload original videos to cloud and use cloud service to get all things done, but offloading task into cloud will result in heavy cellular traffic, which means high energy consumption and expensive communication fee. But if he is a participate in MSON and several video processing services is provided by some nearby mobile devices, he can invoke such mobile services on nearby mobile devices through D2D communications. If these services cannot meet his requirement, several services can be composed. Due to users' mobility, the availability of service can vary, invoking mobile services provided by other users may face new challenges that traditional composition methods cannot handle. Thus, a mobile service composition
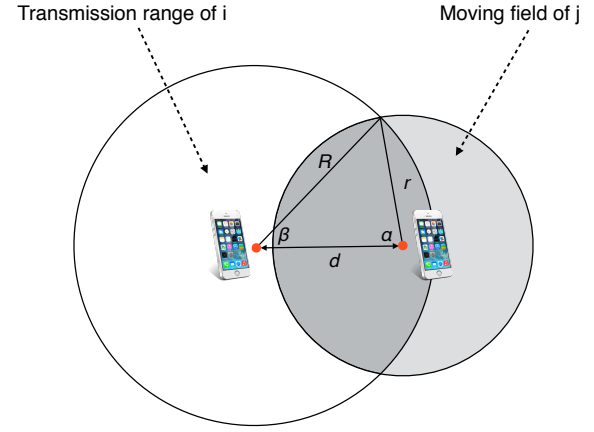
model which can capture mobile services' availability need to be proposed.

4 MOBILE SERVICE COMPOSITION MODEL

4.1 Preliminaries

To facilitate modeling, reasoning, and analysis of the mobile service composition problem, we first present some basic definitions.

Definition 1 (Mobile Service): A mobile service is denoted by a two-tuple $ms = (info, QoS)$, where:

1) $info$ is the description of a mobile service which include service name, functionality, input parameters and response data.

2) $QoS = \{q_{rt}, q_{price}, q_{ava}, ...\}$ is a set of quality attributes, including response time, price, availability, etc.

Definition 2 (MSON participant): A MSON participant is mobile service user who can be a service provider or a requesters. It is denoted by a two-tuple $u = (P, C)$, where:

1) P is the set of mobile services exposed to other MSON participants.

2) C is the set of mobile services discovered from nearby MSON participants.

Definition 3 (Mobile Service Composition Plan): A Mobile Service Composition Plan $mscp = (T, E)$ is modeled as a directed acyclic graph (DAG) where $T = t_1, t_2, ..., t_n$ is the set of tasks and E is the set of directed edges. An edge e_{ij} of the form (t_i, t_j) exists if there is a data dependency between t_i and t_j, case in which t_i is said to be the parent task of t_j and t_j is said to be the child task of t_j. Based on this definition, a child task cannot be executed until all of its parent tasks are completed.

4.2 Mobile Service Availability

In an MSON, the availability of mobile service is variable and dynamically by users' mobility. As illustrated by an example in Fig. 3, there are two mobile users i and j with identical transmission rage R. User i is a mobile service requester while user j a mobile service provider. Each user moves freely and it is assumed that the moving area is a circle with a radius of r in a certain amount of time. d is the distance between i and j. If user j moves outside the transmission range of its neighbouring user i, then user j

is unreachable for user i and consequently the services on user j become unavailable to user i [8].

Suppose a mobile service running on mobile node j is a candidate service for a task requested by user i, and the availability of candidate service (denote as $Ava(i,j)$) equals the probability of user j staying inside the transmission range of user i, and it can be calculated as follow:

$$Ava(i,j) = \frac{S_{i \bigcap j}}{S_j} \tag{1}$$

Where $S_{i \bigcap j}$ is the area of the user j moving field inside the transmission range of user i, S_j is the moving field area of the user j.

We use r, R and d to calculate $S_{i \bigcap j}$ and S_j. The transmission range of a node R is a preset value (e.g., usually 10m for bluetooth and 25m for Wi-Fi). Note that most of wireless transaction protocols have defined the RSSI (Received Signal Strength Indicator), then distance d between mobile user i and user j can be calculated by signal strength. The moving radius of a mobile user r is its moving speed v multiplied by the average service time t. Here t can be statistically calculated as the average value of last n times of service invoke, namely, $t = \Sigma_{i=1}^{n} t_i / n$. The speed of a mobile user v can be get through GPS data or other mobile sensor (e.g., Gyro-sensor), then $r = v \times t$.

Finally, $S_{i \bigcap j}$ can be calculated as follow:

$$S_{i \bigcap j} = [(\frac{2\alpha}{2\pi} \times \pi r^2) - (\frac{r sin\alpha cos\alpha}{2} \times 2)] \tag{2}$$
$$+ [(\frac{2\beta}{2\pi} \times \pi R^2) - (\frac{R sin\beta cos\beta}{2} \times 2)]$$
$$= \alpha r^2 + \beta R^2 - (r^2 sin\alpha cos\alpha + R^2 sin\beta cos\beta)$$

Where

$$\alpha = arccos(\frac{r^2 + d^2 - R^2}{2r \times d}) \tag{3}$$
$$\beta = arccos(\frac{R^2 + d^2 - r^2}{2R \times d})$$

S_j can be calculated by:

$$S_j = \pi r^2 \tag{4}$$
$$= \pi \times (s \times t)^2$$

Therefore, the availability of mobile service between requester i and provider j in a certain time can be calculated as follow:

$$Ava(i,j) = \frac{\alpha r^2 + \beta R^2 - (r^2 sin\alpha cos\alpha + R^2 sin\beta cos\beta)}{\pi s^2 t^2} \tag{5}$$

Mobile service availability can capture user's mobile behavior, and we use it as an important QoS attribute to construct QoS model for service composition in next subsection.

4.3 QoS Model for Mobile Service Composition

For mobile service requesters to select candidate service, QoS must be considered [22], [23], [24]. Generally, QoS attributes include response time, price, reliability, and reputation, we introduce mobile service availability as an important QoS attribute in this paper to describe user's mobility behavior. QoS attributes in this paper can be classified

TABLE 1
Examples of aggregation functions for QoS

Pattern	Resopnse Time	Availability
sequence	$\sum_{i=1}^{n} q_{rt}(S_i)$	$\prod_{i=1}^{n} q_{ava}(S_i)$
parallel	$Max\{q_{rt}(S_i)\}$	$\prod_{i=1}^{n} q_{ava}(S_i)$
choice	$\sum_{j=1}^{n} p_j \times q_{rt}(S_j)$	$\sum_{j=1}^{n} p_j \times q_{ava}(S_j)$
loop	$k \times q_{rt}(S_i)$	$[q_{ava}(S_i)]^k$

into two categories: positive (Q^+) and negative (Q^-). For positive attributes, larger values indicate better performance (e.g., reputation and availability), while for negative attributes, smaller values indicate better performance (e.g., response time and cost).

A service composition instance csi can be represented as $csi = \{s_{(1,i)}, s_{(2,j)}, ..., s_{(n,k)}\}$, where $s_{(j,k)}$ is the selected concrete services. For example, $s_{(1,i)}$ means the i-th service candidate is selected to execute $task_1$. For a csi, its each QoS attribute is determined by its concrete components and orchestration patterns. Table.1 lists the aggregation functions for response time and availability for sequential, loop, choice, and parallel composition patterns. We can find more aggregation functions found in [25] and [26].

In order to facilitate ranking of different composite service instances csi in terms of QoS, we utilize simple additive weighting (SAW) as the QoS utility function to map the QoS value into a real value. SAW first normalizes the QoS attribute values into real values between 0 and 1, through comparison with the maximal and minimal values; then it sums the normalized values multiplied with a preference weight w_t. According to SAW, the QoS utility of a csi can be calculated using e.q (6), where, $q_t(csi)$ is the aggregated value of the type-t QoS attribute of csi, and $q_{t,max}$ and $q_{t,min}$, respectively, denote the maximal and minimal possible aggregated values of the type-t QoS attribute [22].

$$U(csi) = \sum_{q_t \in Q^-} \frac{q_{t,max} - q_t(csi)}{q_{t,max} - q_{t,min}} \times w_t \tag{6}$$
$$+ \sum_{q_t \in Q^+} \frac{q_t(csi) - q_{t,min}}{q_{t,max} - q_{t,min}} \times w_t$$

4.4 Problem Formulation

Base on the above discussion, we can give the definition of the service composition over MSON problem.

Definition 4 (MSON Service Composition): Given a service composition request req by a mobile user u, perceive nearby service and select suitable concrete services to achieve an optimal service composition instance csi with the best QoS, that is

$$maximize \quad : U(csi) \tag{7}$$
$$subject\ to \quad : t_i \in \{1, 2, 3, ..., n\}$$
$$s_i \in \{1, 2, 3, ..., m\}$$

where $U(csi)$ is the objective function mentioned in e.q (6), $t_i \in [1, n]$ is the index of the tasks in the composition plan, $s_i \in [1, m]$ is the index of service candidates for the i-th task.

Theorem 1: The service composition problem over MSON (Definition 6) is NP-hard.

Proof: We can reduce the service composition problem over MSON problem to a knapsack problem, and this problem can be solved by integer programming. The canonical form of integer linear program to search the optimal solution of this problem can be expressed as follow [27]:

$$maximize \quad : F(X) = c^T X \qquad (8)$$
$$subject \ to \quad : Ax \le b,$$
$$x_i \in \mathbb{Z}^n$$
$$x_i > 0$$

Where $F(x)$ is objective function, X is feasible solution, n is a positive integer, b and c are vectors, A is a matrix. For the problem of selecting optimal services composition over MSON, the vector $X = \{s_{(1,i)}, s_{(2,j)}..., s_{(n,k)}\}$ can describe a feasible solution as a service composition with n tasks. The optimal solution X^* satisfies the following conditions:

1) X^* is a feasible solution.

2) for all feasible X, $F(X^*) \le F(X)$.

The target of the mobile service composition problem over MSON is to find the biggest $F(x)$. Thus, the problem is equivalent to the integer programming problem which is known to be NP-hard. Then the service composition problem over MSON is NP-hard.

5 THE KH-BASED ALGORITHM FOR MOBILE SERVICE COMPOSITION

For the problem we formulate above, integer programming can be utilized to obtain the optimal solution. However, integer programming might cost much more time with the increment of problem size because of its poor scalability [28]. To solve this problem in polynomial time, an meta-heuristic algorithms such as GAs and PSO, can be utilized to find the near optimal solution.

Krill-Herd algorithm [29] is new generic stochastic optimization approach for the global optimization problem which is inspired by predatory behavior and communication behavior of krill. In this section, we will introduce a Krill-Herd based algorithm to solve the problem of mobile service composition over MSON.

5.1 Encoding

In KH, composite service instance is encoded as a krill individual, the krill individual with the best position corresponds to the optimal mobile service composition. The target of algorithm is to find the krill individual with the best position, which means to find the best mobile service composition with the best QoS utility. Therefore, once the optimal krill individual is found, the best mobile service composition is obtained.

In this paper, the position vector of each krill individual is represented by an integer array with its length equal to the number of involved tasks. The i-th entry in the array, in turn, refers to the selection result of the task t_i . That is to say, given that the value of the n-th entry is k, it indicates that $s_{(n,k)}$ is the selected concrete service to execute t_n. Fig. 4 illustrates this krill encoding.
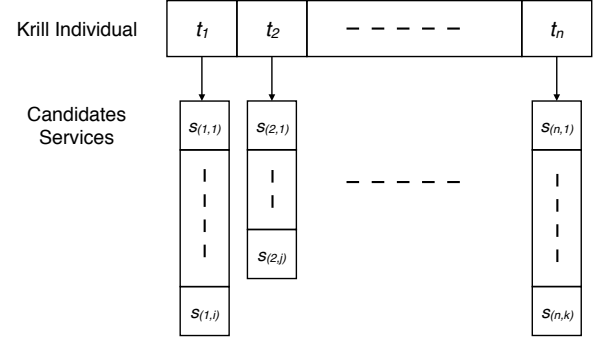

Fig. 4. Krill encoding

5.2 Motion operator

Motion operator is the key component of KH algorithm. As shown in e.q (10), the position of each krill individual is determined by three main factors: 1) motion influenced by other krill; 2) foraging action; 3) physical diffusion.

$$\frac{dX_i}{dt} = N_i + F_i + D_i \qquad (9)$$

where individual $X_i = \{ms_{(1,j)}, ms_{(2,k)}, ..., ms_{(n,l)}\}$ represents the i-th composition service instance in population, n is the number of tasks in the service composition, N_i, F_i, and D_i denote the motion influenced by other krill individuals, the foraging motion, and the physical diffusion, respectively.

1) Movement induced by other krill individuals

The motion induced by other krill individuals N_i means to learn from neighbor mobile service compositions. It can be formulated as follow:

$$N_i^{new} = N_{max}\alpha_i + \omega_n N_i^{old} \qquad (10)$$

where

$$\alpha_i = \alpha^{target} + \alpha^{local} \qquad (11)$$

α_i is the direction of the induced motion and it can be evaluated by target swarm density (target effect α^{target}), local swarm density (local effect α^{local}). N_{max} is the maximum induced speed, $\omega_n \in [0, 1]$ the inertia weight of the induced motion, N_i^{old} is the last induced motion influenced by other krill individuals.

2) Foraging Motion

Similarly, the foraging motion F_i is to learn from the current optimal composite service instance. F_i has two parts: the current food location and the information about the previous location. For the individual X_i, we can formulate this motion as follow

$$F_i = V_f\beta_i + \omega_f F_i^{old} \qquad (12)$$

where

$$\beta_i = \beta_i^{food} + \beta_i^{best} \qquad (13)$$

where V_f is the foraging speed, $\omega_f \in [0, 1]$ is the inertia weight of foraging, and F_i^{old} is the last foraging motion. β_i is the direction of the foraging motion.
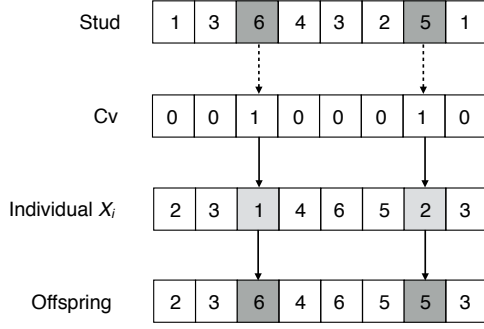
3) Random diffusion

Fig. 5. Crossover operator

For individual X_i, the physical diffusion is considered to be a random process. This motion includes two components: a maximum diffusion speed and a random directional vector, it can be formulated as follow

$$D_i = D_{max}\delta \qquad (14)$$

where D_{max} is the maximum diffusion speed and $\delta \in [-1, 1]$ is a random directional vector.

5.3 Stud selection and crossover operator

The crossover operator plays an important role in Genetic algorithm for global optimization, we use this operator in KH algorithm to enhance the search capability. The crossover operator in this paper is controlled by a dynamic crossover rate C_{rate} which can be obtain as follow

$$C_{rate} = Cr + (1 - Cr) \times \frac{U_{best} - U_i}{U_{best} - U_{worst}} \qquad (15)$$

Where C_r is a pre-set fixed crossover rate, U_i is the i-th individual's QoS utility, U_{best} is the current best QoS utility value, similarly, U_{worst} is the current worst QoS utility value.

Then we can use C_{rate} to generate i-th individual's crossover vector $Cv = \{c_1, c_2, ..., c_n\}$, it can be manipulated as follows

$$c_i = \begin{cases} 1, & if \ rand(0, 1) < C_{rate} \\ 0, & else \end{cases} \qquad (16)$$

Inspired by SGA [30] (a type of GA which employs the optimal genome for crossover at each generation), we introduce stud selection procedure to improve KH's search capability. From algorithm 1, we can see that for each individual X_i to crossover, we choose the optimal individual $Stud$ (i.e., the individual with highest QoS utility value) to mating. As shown in Fig. 5, the characteristics from individual $Stud$ are copied to individual X_i according to crossover vector Cv.

5.4 Update position

After crossover, the offspring should be evaluated and updated to current evolutionary sequence. According to the three motion actions, the time-relied position from time t and Δt can be formulated by the following equation

$$X_{i+1} = X_i + \Delta t \frac{dX_i}{dt} \qquad (17)$$

Algorithm 1 Crossover operation

Input: Population X; Individual X_i to crossover; The number of tasks $taskNumber$;
1: Sort all krill individuals in population X by its QoS utility, get optimal individual $Stud$, save the best QoS utility value as U_{best} and the worst QoS utility value as U_{worst}
2: $C_{rate} \leftarrow$ calcCrossoverRate$(X_i, U_{best}, U_{worst})$
3: **for** $i = 0$ to $taskNumber$ **do**
4:    $r \leftarrow rand(0, 1)$
5:    **if** $r < C_{rate}$ **then**
6:       $Cv[i] \leftarrow 1$
7:    **else**
8:       $Cv[i] \leftarrow 0$
9:    **end if**
10: **end for**
11: **for** $i = 0$ to $taskNumber$ **do**
12:    $X_i[i] \leftarrow X_i \wedge (1 - Cv[i]) + Stud \wedge Cv[i]$
13: **end for**

---

where

$$\Delta t = C_t \sum_{j=1}^{n}(UB_j - LB_j) \qquad (18)$$

where n is the tasks number of composition service, UB_j and LB_j are upper and lower bounds of candidate services for the j-th task, respectively. C_t is a constant value to scale the searching space and we set it to $1/2n$ in this paper. Finally, the overall KH algorithm process can be describe in Algorithm 2.

Algorithm 2 KH algorithm

Input: Number of population size PS, Number of max iteration MI;
1: Generate initial population as $X = (X_1, X_2, ..., X_{PS})$
2: Evaluate the QoS utility value of each krill individual in X
3: **for** $i = 0$ to MI **do**
4:    **for** $i = 0$ to PS **do**
5:       $X_i^{'} \leftarrow motionOperator()$
6:       $X_i^{''} \leftarrow crossoverOperator(X_i^{'})$
7:       $U_i^{'} \leftarrow evaluateQoSutility(X_i^{'})$
8:       $U_i^{''} \leftarrow evaluateQoSutility(X_i^{''})$
9:       **if** $U_i^{''} < U_i^{'}$ **then**
10:          update position by equation (18) as X_{i+1}
11:       **else**
12:          accept $X_i^{''}$ as X_{i+1}
13:       **end if**
14:    **end for**
15: **end for**
16: Output the best solution

---

6 SIMULATION AND EVALUATION

In this section, we first discussed the experimental environment settings, and then the KH-based approach for the mobile service composition algorithm is evaluated from the perspective of optimality and scalability, respectively.

TABLE 2
User u 's D2D contract traces

Time	Available service provider
t1	Rabbit, Tony, S10, BlueRadios, NORTHOLT
t2	Tony, S10, Rabbit, NORTHOLT, BlueRadios
t3	Rabbit, NORTHOLT, BlueRadios, S10, Tony, Henrymobile, S4
t4	Tony, NORTHOLT, BlueRadios, S10, Rabbit, S4
t5	BlueRadios, S4, AliKatz, NORTHOLT, Rabbit, S25, S10
t6	S25, S10, NORTHOLT, BlueRadios, Rabbit
...	...

TABLE 3
Services exposed by provider

Service Provider	Exposed Service
AliKatz	ms_1, ms_2, ms_3, ms_4
BlueRadios	ms_1, ms_5
Henrymobile	ms_2, ms_4
NORTHOLT	ms_4, ms_5, ms_6
Rabbit	ms_1, ms_4
S4	ms_1, ms_2
S10	ms_6, ms_7
S25	ms_4, ms_5
Tony	ms_1, ms_4
...	...

TABLE 4
Available candidates

Time	Available service
t1	$ms_1^{(3)}, ms_4^{(3)}, ms_5^{(2)}, ms_6^{(1)}, ms_7^{(1)}$
t2	$ms_1^{(3)}, ms_4^{(3)}, ms_5^{(2)}, ms_6^{(2)}, ms_7^{(1)}$
t2	$ms_1^{(4)}, ms_2^{(2)}, ms_4^{(4)}, ms_5^{(2)}, ms_6^{(2)}, ms_7^{(1)}$
t4	$ms_1^{(4)}, ms_2^{(1)}, ms_4^{(3)}, ms_5^{(2)}, ms_6^{(2)}, ms_7^{(1)}$
t5	$ms_1^{(4)}, ms_2^{(2)}, ms_3^{(1)}, ms_4^{(4)}, ms_5^{(3)}, ms_6^{(2)}, ms_7^{(1)}$
t6	$ms_1^{(2)}, ms_4^{(3)}, ms_5^{(3)}, ms_6^{(2)}, ms_7^{(1)}$
...	...

TABLE 5
parameters configuration

PS	MI	CR	V_f	N_{max}	D_{max}
1~60	50	0.6	0.8	0.3	0.2
20	10~150	0.6	0.8	0.3	0.2
20	50	0.01~1.00	0.8	0.3	0.2
20	50	0.6	0.01~3.00	0.3	0.2
20	50	0.6	0.01	0.01~2.00	0.2
20	50	0.6	0.01	0.3	0.01~3.00

6.1 Simulation Setting

To evaluate the optimality and scalability of the proposed approaches, the experiment is run on a personal computer with an Intel Core i5 CPU with 2.4 GHz, 4 GB RAM, macOS and Matlab R2015b Edition.

Since we can not find available realistic datasets which involving both user D2D contact traces and quality of mobile service so far, we attempt to simulate the scenarios for mobile services provision by integrating realistic user D2D contact traces with quality of Web service datasets.

We consider MIT Reality dataset [31] as user D2D contact traces, where user location, Bluetooth devices in proximity, application usage, and phone status (such as charging and idle) were collected from 100 users over several months. This dataset can really reflect diverse network scenarios.

The publicly available quality of Web service (QWS) dataset [32] can be used to characterize the service candidates. This dataset consists of 4500 Web services from 142 users over 64 different time slices (at 15-minute interval) and each QoS data includes two measurements (response time and throughput).

Table 2 is part of D2D contract traces in MIT Reality dataset. For example, there there are five nearby devices within D2D transmission distance at time $t1$ and these devices can be regarded as MSON participant who provision mobile services. Table 3 shows MSON participants and the services they exposed to nearby devices. These mobile service are random chosen from QWS dataset. Table 4 is the Cartesian product of Table 2 and Table 3, it shows how many kinds of services user can exploit at a certain time and how many candidates for each kind of service (i.e., task). For

example, there are five kinds of service available at time $t1$ and there are three candidates for task t_1, three candidates for task t_4, two candidates for task t_5, one candidate for task t_6 and one candidate for task t_7.

6.2 Impact of Parameters

There are six parameters can be adjusted to improve the IKH's performance: population size PS, maximum iteration number MI, crossover rate Cr, foraging speed V_f, maximum induced speed N_{max} and physical diffusion speed D_{max}. As shown in Table 5, we generate six groups of parameters configuration to evaluate the impact of each parameter. For each group of parameters configuration, we tune one parameter and fix the other parameters. For each configuration setting, the IKH algorithm is executed 50 times independently and the average performance was recorded.

Fig. 5(a) shows the impact of population size, we observe that with the increase of population size, the average QoS utility of IKH significantly improved before $PS = 10$, and no significant improvement is observed after population size over 20. Therefore, an excessively large population size (e.g., $PS = 50$) has limited impact on the performance of IKH, and it will result in computing resources waste and high time cost. Similarly, Fig. 5(b) shows that the value of QoS utility significantly increased for higher number of iteration times until to a limit: $MI = 30$. Fig. 5(c) shows the impact of the crossover rate Cr. The performance of KH increases with Cr firstly, then decrease, the best performance is achieved for $Cr = 0.6$. Fig. 5(d) shows the impact of the foraging speed V_f. The performance of IKH increases with V_f firstly, then decrease, the best performance is achieved for $V_f = 0.8$. Fig. 5(e) shows the impact of the induced
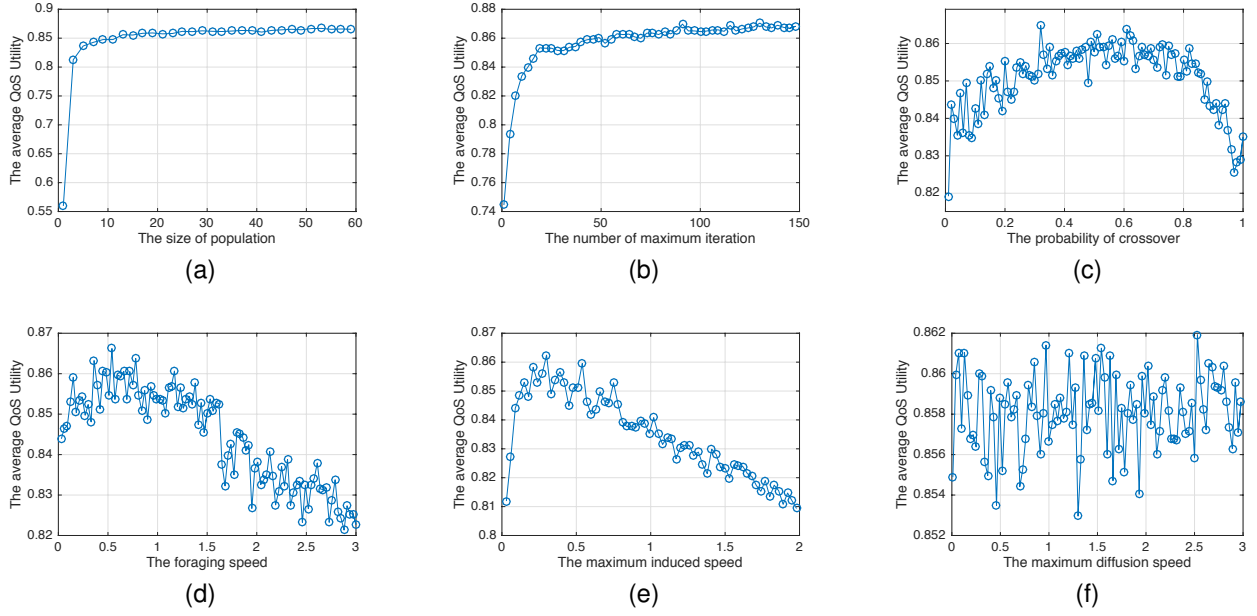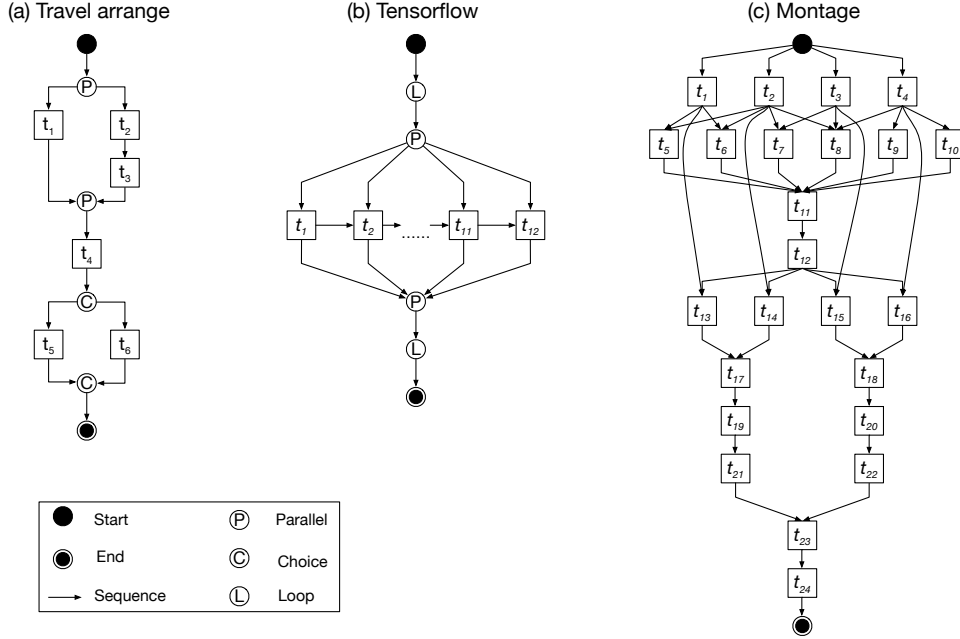
Fig. 6. Impact of parameters.

Fig. 7. The composition plan for case study.

speed N_{max}. The performance of IKH increases with N_{max} firstly, then decrease, the best performance is achieved for $N_{max} = 0.3$. Fig. 5(f) shows the impact of the physical diffusion speed D_{max}. It shows that with the increase of D_{max} the performance of IKH fluctuation irregularly and slightly, without loss of generality, we random generate D_{max} from $[0.2 \sim 0.5]$ in this paper.

6.3 Case study

In this section, we present a case study of different service composition plan in mobile environment, to compare traditional mobile service composition approaches with our proposed framework. Fig. 9 shows three mobile service composition plans for case study. Fig. 9(a) is a well known composition plan for booking tickets [33], it has 6 tasks. Fig. 9(b) is a simple workflow with 12 tasks for Tensorflow [3], Tensorflow is a heterogeneous distributed system for machine learning and it already can be deployed in mobile devices. Fig. 9(c) is a scientific workflow with 24 tasks for Montage. Montage is an astronomical image mosaic engine, it can be used for simulating some picture edit application in mobile phone. We use these three kinds of composition plans to represent different meaningful service composition with different tasks.
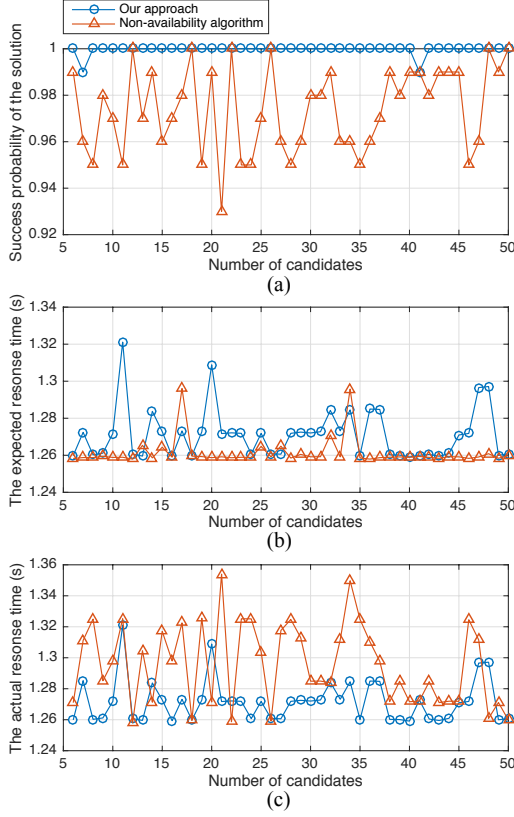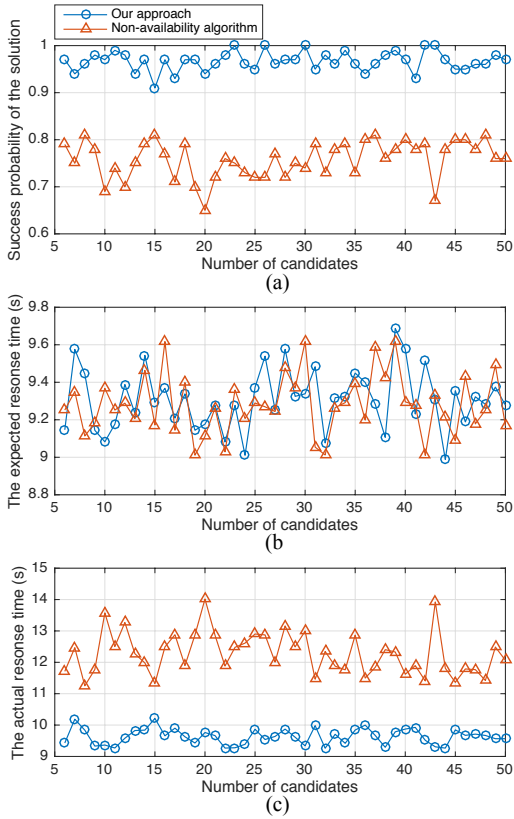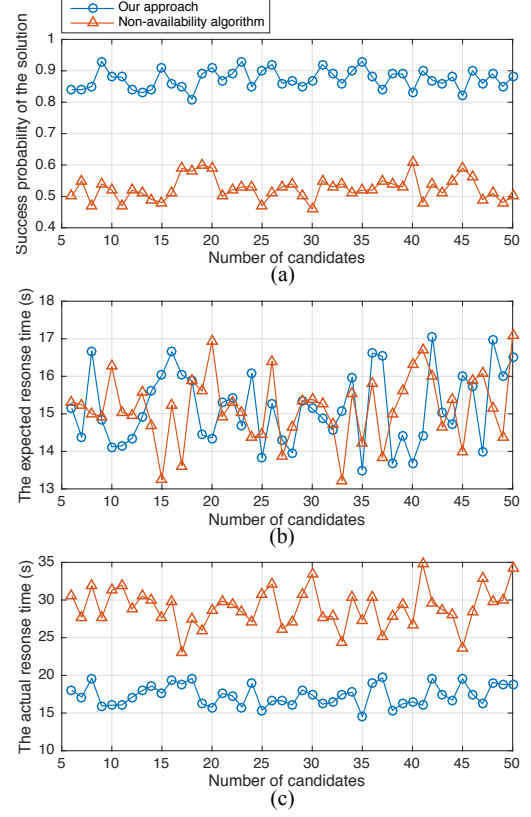
Fig. 8. Case I

Fig. 9. Case II

Fig. 10. Case III

As shown by Fig. 7(a), Fig. 8(a) and Fig. 9(a), our proposed method achieves higher success probability (by almost 100for Case I, average 95for Case II, and average 90for Case III). From Fig. 7(b), Fig. 8(b) and Fig. 9(b) we can see, the solutions made by traditional mobile service composition approach which doesn't consider mobile service availability some times perform better than our method (especially in Case I). That is because our method consider both response time and service availability as optimization Object and in most of time our method can get the same performance as their approach. But in real mobile environment, low success probability means more time cost. A successful service composition response time is shown in Fig. 7(c), Fig. 8(c) and Fig. 9(c), it indicates that our method perform better in actual mobile environment.

Intuitively, the disadvantage of a non-availability approach lies in that it ignores mobility characteristic of mobile users. It therefore tends to choose the best provider who's QoS data is outstanding but will be not available in few second.

7 CONCLUSIONS AND FURTHER STUDIES

In this paper, we propose a comprehensive framework for optimal mobile service compositing on mobile environment. We present a mobile service opportunistic network model (MSON) that fully integrates human mobility behavior factors for mobile service provisioning and introduce a QoS model for mobile service composition. Then we formulate the mobile service composition over MSON problem as an optimal problem to maximize the quality of service

composition, propose a Krill-herd based algorithm to solve it, and a case study based on real-world opportunistic network and some well-known web service dataset show that our proposed approach outperforms current standard composition methods in mobile environments.

We plan to consider the following topics for future work: 1) Some prediction methods (e.g., hidden Markov model and neural networks) can be used to predict user future's movement to formulate a better user mobility model; 2) more QoS metrics (e.g., service price and service reputation) are supposed to be analyzed and blends into our QoS model; 3) this work doesn't consider Service-Level-Agreement (SLA) constrains. We intend to consider SLA constraints and introduce corresponding algorithms to generate better composition.

REFERENCES

[1] M. Satyanarayanan, "Mobile computing: the next decade," in *Proceedings of the 1st ACM workshop on mobile cloud computing & services: social networks and beyond*. ACM, 2010, p. 5.

[2] E. Bastug, M. Bennis, M. Médard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 110–117, 2017.

[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[4] 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*, vol. 13, no. 18, pp. 1587–1611, 2013.

[5] R. Balani, "Energy consumption analysis for bluetooth, wifi and cellular networks," *Online Httpnesl Ee Ucla Edufwdocumentsreports2007PowerAnalysis Pdf*, 2007.

[6] S. Giordano and D. Puccinelli, "The human element as the key enabler of pervasiveness," in *Ad Hoc Networking Workshop (Med-Hoc-Net), 2011 The 10th IFIP Annual Mediterranean*. IEEE, 2011, pp. 150–156.

[7] S. Deng, L. Huang, H. Wu, W. Tan, J. Taheri, A. Y. Zomaya, and Z. Wu, "Toward Mobile Service Computing: Opportunities and Challenges," *IEEE Cloud Computing*, vol. 3, no. 4, pp. 32–41, 2016.

[8] K. Yang, A. Galis, and H.-H. Chen, "Qos-aware service selection algorithms for pervasive service composition in mobile wireless environments," *Mobile Networks and Applications*, vol. 15, no. 4, pp. 488–501, 2010.

[9] C. Zhang, L. Zhang, and G. Zhang, "Qos-aware mobile service selection algorithm," *Mobile Information Systems*, vol. 2016, 2016.

[10] J. Wang, "Exploiting mobility prediction for dependable service composition in wireless mobile ad hoc networks," *IEEE Transactions on Services Computing*, vol. 4, no. 1, pp. 44–55, 2011.

[11] M. Conti and S. Giordano, "Mobile ad hoc networking: milestones, challenges, and new research directions," *IEEE Communications Magazine*, vol. 52, no. 1, pp. 85–96, 2014.

[12] O. Turkes, H. Scholten, and P. J. Havinga, "Cocoon: A lightweight opportunistic networking middleware for community-oriented smart mobile applications," *Computer networks*, vol. 111, pp. 93–108, 2016.

[13] C. Fortuna and M. Mohorcic, "Dynamic composition of services for end-to-end information transport," *IEEE Wireless Communications*, vol. 16, no. 4, 2009.

[14] L. Pu, X. Chen, J. Xu, and X. Fu, "Crowd foraging: A qos-oriented self-organized mobile crowdsourcing framework over opportunistic networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 4, pp. 848–862, 2017.

[15] S. Deng, L. Huang, J. Taheri, J. Yin, M. Zhou, and A. Y. Zomaya, "Mobility-aware service composition in mobile communities," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 3, pp. 555–568, 2017.

[16] Y. Li and W. Wang, "Can mobile cloudlets support mobile applications?" in *Infocom, 2014 proceedings ieee*. IEEE, 2014, pp. 1060–1068.

[17] W. Chang and J. Wu, "Progressive or conservative: Rationally allocate cooperative work in mobile social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 7, pp. 2020–2035, 2015.

[18] G. S. Tuncay, G. Benincasa, and A. Helmy, "Participant recruitment and data collection framework for opportunistic sensing: a comparative analysis," in *Proceedings of the 8th ACM MobiCom workshop on Challenged networks*. ACM, 2013, pp. 25–30.

[19] J. Wu, M. Xiao, and L. Huang, "Homing spread: Community home-based multi-copy routing in mobile social networks," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 2319–2327.

[20] C. Jiang, L. Gao, L. Duan, and J. Huang, "Exploiting data reuse in mobile crowdsensing," in *Global Communications Conference (GLOBECOM), 2016 IEEE*. IEEE, 2016, pp. 1–6.

[21] S. Liu and A. D. Striegel, "Exploring the potential in practice for opportunistic networks amongst smart mobile devices," in *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 2013, pp. 315–326.

[22] Q. Wu, F. Ishikawa, Q. Zhu, and D.-H. Shin, "QoS-Aware Multi-granularity Service Composition: Modeling and Optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 11, pp. 1565–1577, 2016.

[23] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, 2014.

[24] X. Luo, M. Zhou, Y. Xia, Q. Zhu, A. C. Ammari, and A. Alabdulwahab, "Generating highly accurate predictions for missing qos data via aggregating nonnegative latent factor models," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 3, pp. 524–537, 2016.

[25] M. C. Jaeger, G. Rojec-Goldmann, and G. Muhl, "Qos aggregation for web service composition using workflow patterns," in *Enterprise distributed object computing conference, 2004. EDOC 2004. Proceedings. Eighth IEEE International*. IEEE, 2004, pp. 149–159.

[26] H. Zheng, W. Zhao, J. Yang, and A. Bouguettaya, "Qos analysis for web service compositions with complex structures," *IEEE Transactions on Services Computing*, vol. 6, no. 3, pp. 373–386, 2013.

[27] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[28] G. L. Nemhauser and L. A. Wolsey, "Integer programming and combinatorial optimization," *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, vol. 20, pp. 8–12, 1988.

[29] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.

[30] W. Khatib and P. J. Fleming, "The stud ga: a mini revolution?" in *International Conference on Parallel Problem Solving from Nature*. Springer, 1998, pp. 683–691.

[31] N. Eagle and A. S. Pentland, "Reality mining: sensing complex social systems," *Personal and ubiquitous computing*, vol. 10, no. 4, pp. 255–268, 2006.

[32] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating qos of real-world web services," *IEEE Transactions on Services Computing*, vol. 7, no. 1, pp. 32–39, 2014.

[33] Q. Wu and Q. Zhu, "Transactional and qos-aware dynamic service composition based on ant colony optimization," *Future Generation Computer Systems*, vol. 29, no. 5, pp. 1112–1119, 2013.