Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090640

Location-based Service Composition Algorithm in a Wireless Ad hoc Network

Songqiao Han[1,2], *Xiaoling Hao*[1,2], *Hailiang Huang*[1,2,*], *Songxin Wang*[1,2] *and Yong Zhang*[1,2]

[1] School of Information Management and Engineering, Shanghai University of Finance and Economics, Shanghai 200433, P. R. China
[2] Key Laboratory of Financial Information Technology of Shanghai City, Shanghai 200433, P. R. China

Received: 6 Mar. 2015, Revised: 4 May 2015, Accepted: 5 May 2015
Published online: 1 Nov. 2015

Abstract: Location is increasingly becoming a crucial criterion of Quality of Service (QoS) in mobile computing environments due to the widespread adoption of location-sensing devices. Although some previous work has studied the service composition protocols based on QoS in ad hoc networks, they do not explicitly take services' location into account. In this paper, we present a distributed service composition algorithm suitable for ad hoc networks to find the optimal composite service with the lowest cost while satisfying distance constraint. Based on the idea of a source-initiated on-demand protocol, a transmission approach of service request messages is exploited to dynamically discover and compose the appropriate basic services in a service network. To solve the broadcast storm problem of control messages, a message filtering algorithm is proposed to effectively discard the unqualified control messages. Simulation results demonstrate that the proposed algorithm significantly outperforms the traditional algorithms in terms of success rate, cost and control message overhead.

Keywords: Service composition, location-based service, ad hoc networks

1 Introduction

Technological advances in mobile device design and development as well as wireless networking are leading us towards a vision of pervasive computing where a large number of smart devices are around users and provide an array of services. A group of wireless devices can form an ad hoc network without the aid of any existing network infrastructure. Due to resource limitation and the nature of heterogeneity, the device nodes generally provide small and simple services some of which may perform similar or identical functionality, but possibly distributed in different locations. Thus, users often need to discover and compose some desired services within a limited location or distance to accomplish their complicated tasks. For example, there are a variety of restaurants, hotels, shops, theaters, parking services and supermarkets in a city. They exploit wireless devices to broadcast their services towards the potential users in their vicinities, at the same time the wireless devices can form an ad hoc network. When visitors arrive at the city, they use their smart phones or pads to search the desired services, dynamically compose them, and finally obtain the satisfied composite services with the minimal cost within a limited distance. Similar examples in ad hoc networks rang from mobile commerce environments to army warfront environments, sensor networks to deep space exploration research [1].

Service composition has been widely researched in the areas of Service Oriented Architecture (SOA) and Service Oriented Computing (SOC) [2]. In addition, plenty of works on QoS-aware service composition in wired fixed networks have been conducted to compose a set of appropriate services into a richer service satisfying certain application-level Quality of Service (QoS) requirements [3–5]. However, these service composition approaches cannot directly applied in wireless ad hoc networks with intermittent connectivity and the absence of central servers. Thus, some work is carried out to study the protocols and infrastructures of service discovery and service composition to satisfy functional requirements of users in ad hoc networks [1, 6, 7]. Considering the non-functional properties of composite services, some researches are conducted to investigate QoS aware service composition middleware and algorithms. In addition to application level QoS criteria, such as delay, price, availability, reputation and reliability, they concern about

* Corresponding author e-mail: hlhuang@shufe.edu.cn

the network level QoS criteria, including node availability, network delay and network reliability. More recently, some researchers studied the minimum service disruptions [8] and predicted the mobility of service providers [9] to obtain dependable service composition. However, these works do not explicitly consider location based service discovery and service composition in ad hoc networks.

Location has been one of crucial QoS criteria in mobile computing environments. In general, users prefer to search and use the services within a specified area around them or a limited distance apart from them. Many location based services have been easily accomplished based on outdoors positioning technologies, such as GPS, and indoors positioning technologies [10], such as WiFi or RFID positioning. In this paper, we formulate the location aware service composition problem in an ad hoc network, and transform the problem into a service path discovery within a limited distance in a service network. Then we present a distributed service composition algorithm to discover desired basic services and compose them to obtain a composite service with satisfying distance constraints while minimizing the cost, such as price and resource consumptions. This paper mainly considers services' distance rather than other QoS metrics, such as delay, availability, reliability and reputation. Actually, the proposed algorithm can be used to deal with the service composition based on the other QoS metrics. We focus on the sequential service composition pattern, which is the basic of more complicated structures with the loop, selective or parallel branches.

The remainder of this paper is organized as follows: Section 2 reviews the related work; Section 3 formulates the service composition problem for ad hoc networks; Service composition algorithm is proposed in Section 4; Section 5 conducts simulation experiments to evaluate the performances of our algorithm, and finally the paper concludes in Section 6.

2 Related Work

Although service composition has been widely studied for wired-infrastructure environments like Internet [3, 11], its approaches cannot be directly applied in the ad hoc networking environments where it is difficult to find such an infrastructure to discover and compose services in a centralized way. Some work studied service composition infrastructures and frameworks in ad hoc networks. Gaber et al. [7] utilized a group of autonomous agents to represent composite services and established affinity relationships among agents to achieve service discovery and composition in pervasive computing. Chen et al. [6] presented a P2P mobile service composition infrastructure based on context awareness. Chakraborty et al. [1] elegantly investigated service composition in mobile environments and presented a distributed service composition protocols based on distributed brokerage mechanisms. Basu et al. [12] described a hierarchical task-graph based approach to enable service composition in ad hoc networks. However, these researches used best-effort approaches and do not concern about QoS and resource consumptions that is important for real-time systems.

More recently, some work focused on the QoS aware service composition in ad hoc networks. Park et al. [13] proposed a SOA-based middleware to support QoS control of mobile applications and conserve energy consumption. Yang's work [14] proposed a QoS model specifically for pervasive services with consideration of user-perceived factors, such as availability, reliability, price and delay, and mobile wireless work characteristics, such as node availability, network delay and network reliability. Mokhtar [8] presented a solution of a QoS aware service composition in pervasive computing environments with the aid of QoS-aware semantic service discovery and QoS-aware integration of service conversations. Kumar et al. proposed an ant colony based service selection algorithm with respect to jitter, hop count and service availability in wireless mesh networks environment [15]. Luo et al. [16] presented a network-aware algorithm for service composition with consideration of network availability, delay, price and reputation in wireless environments. In addition, some work has been conducted to obtain reliable composite services in this field. A service composition and recovery framework, including service routing and network routing, is conducted to achieve minimum service disruptions for mobile ad hoc networks in the work [17]. Wang [9] predicted the mobility of service providers to achieve dependable service composition in mobile ad hoc networks.

In all, the existing work mostly focused on the service composition middlewares, service composition based on QoS, and reliable service composition methods. To our knowledge, however, there is no research work to study explicitly location aware service composition in ad hoc networks. In this paper, we are to utilize control message transmission mechanisms to discover and compose services based on distance constraint in order to minimize the cost of composite services.

3 Notation and Formalization

3.1 Service Composition Request

A service composition request mainly consists of a function graph G^F and distance requirement D_{req}. The function graph is a directed graph with nodes representing elementary functions and edges representing the dependency links between functions. The dependency link indicates that the output of one function is used as the input by its successor. For simplification, this paper only focuses on discussing the linear function graph whose solution is the basic of that of more complicated function graphs.

Definition 1 (*Linear function graph*). An linear function graph is a sequence of functions $[f_1, f_2, \dots, f_m]$, such that f_1 is the first function, f_m is the final function, and for every function f_i $(1 < i < m)$, its direct predecessor is f_{i-1} and its direct successor is f_{i+1}. The distance requirements refer to the maximum allowed distance of service compositions which will be discussed in Section 3.3.

3.2 Service Network

In ad hoc networks, there are lots of device nodes some of which provide various services with different functions or different QoS criteria, and others of which do not provide any service, and only participate in service composition by forwarding the services for other nodes.

In such a network, there are three types of entities: devices, links and services, which form a service network. Nodes represent devices and edges represent the wireless links between two devices. Services offered by devices refer to self-constrained application units offering certain functionalities. A service composition aims to connect the needed component services on different nodes to provide users a unified service. For a composite service, nodes are classified into three categories: service node, relay node and dumb node. A service node is the node offering at least one component service for the composite service; a relay node only transmits the messages as an intermediate node; and a dumb node provides neither basic services nor control message transmissions of the composite service.

3.3 Distance and Cost

The distance of a composite service is the sum of Euclidean distance between each two adjacent service nodes. Suppose that a composite service includes m service nodes that are denoted by $< SN_1, SN_2, \dots, SN_m >$ in order, and the coordinates of the i-th node SN_i are denoted by (x_i, y_i). So the distance D_S of the composite service S can be expressed by formula (1).

$$D_S = \sum_{i=0}^{m-1} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \tag{1}$$

The cost of a service refers to the price of the service or its resource consumptions such as the usage of computation, network bandwidth, energy and time. In this paper, we use the price of a service as its cost. If considering resource consumptions, the cost of a service can be obtained using the cost functions of the resources. The cost of a composite service is the sum of the cost of its all basic services. Suppose that a composite service includes n basic services s_i $(i = 1, 2, \dots, n)$, and the cost of service s_i is denoted by c_i. Then the cost C of the composite service S can be expressed by formula (2).

$$C_S = \sum_{i=1}^{n} c_i \tag{2}$$

3.4 Problem Formulation

We formulate a location-aware service composition problem in a wireless ad hoc network as a service path discovery problem in a service network. That is, a service composition is to search a service path that sequentially connects the service nodes performing functions of a function graph. More specific definition of a service path is given as follows.

Definition 2 (*Service Path*). A set of pairs $p_s = < N_1, S_1 >, < N_2, S_2 >, \dots, < N_n, S_n >$ is a service path of function graph $G^F = [f_1, f_2, \dots, f_m]$ iff:

In a set N of nodes $[N_1, N_2, \dots, N_n]$, node N_1 is source node of the path, node N_n is destination node of the path. For each node $N_i (1 < i < n)$, it can directly communicate with both its predecessor N_{i-1} and its successor N_{i+1}. A set of links $l_{ij} (1 \le i < n, j = i+1)$ between N_i and N_j is denoted by L.

Logical service S_i is the service that is used by path p_s on node N_i. If N_i is a relay node, its service $S_i = 0$ since N_i does not provide any service. If N_i is a service node, its service $S_i \neq 0$ since N_i offers one or more actual services for path p_s.

From a set S_l of the logical services $[S_1, S_2, \dots, S_n]$, we select service $S_i (S_i \neq 0)$ in order and form a set S_a of actual services $[s_1, s_2, \dots, s_m] (m \le n)$ where service s_i can perform function f_i, which is denoted as $s_i \sim f_i$.

For example, in Fig. 1, the solid line represents a service path $p_s = \{< N_a, 0 >, < N_b, 0 >, < N_c, s_1 >, < N_d, 0 >, < N_e, s_2 >, < N_f, 0 >, < N_g, 0 >, < N_h, s_3 >\}$ and node set $N = [N_a, N_b, N_c, N_d, N_e, N_f, N_g, N_h]$, logical service set $S_l = [0, 0, s_1, 0, s_2, 0, 0, s_3]$ and actual service set $S_a = [s_1, s_2, s_3]$. Note that when a node provides two consecutive services for p_s, the two nodes corresponding to the two services are same in p_s.

Suppose the distance of a service path p be D_p, and its cost be C_p. The service path p which can perform G^F and satisfy $D_p \le D_{req}$ is defined as a feasible service path p^f, which can be denoted as $p^f \rightarrow (G^F, D_{req})$. In a service network, there may exist multiple feasible service paths $P^f = \{p_1^f, p_2^f, \dots, p_r^f\}$. Hence, the objective of our algorithm is to find the optimal service path p_{opt}^f which is the feasible service path with the minimal cost. The optimal service path can be expressed as follows:

$$p_{opt}^f = \arg \min_{p_i^f \in P^f} C_p(p_i^f) \tag{3}$$

4 Service Composition Algorithm

Based on main ideas of a source-initiated on-demand protocol, a source node creates request messages of service composition and transmits them in an ad hoc network. The messages discover and compose dynamically the appropriate basic services on the visited nodes in order. The discovered feasible service paths are returned to the source node where the best service path with the minimal cost is selected as the final composite service.

4.1 Service Composition Requests

A user requires a composite service that can accomplish graph function $G^F = [f_1, f_2, \ldots, f_m]$ under the condition of satisfying distance requirement D_{req}. Then a client node or source node, such as user's smart phone, is assigned to initiate a service path discovery by originating a Service Composition Request (SREQ) message and broadcasting it in a service network. The structure of SREQ message header can be expressed as (*msgId*, *sourceNode*, *preServiceNode*, *nextFunction*, *sumDistance*, *sumCost*, *maxDistance*, *functionGraph*, *passedPath*). The *msgId* is the identification number of the SREQ on the source node. The *sourceNode* is the address of the node initiating this message. The *preServiceNode* represents the last service node in service path p_{tr}. The *nextFunction* is the function to be discovered in G^F. The *sumDistance* and *sumCost* are the distance and cost of p_{tr}, respectively. The *maxDistance*, *functionGraph* and *passedPath* represent D_{req}, G^F and p_{tr}, respectively. A SREQ message can be uniquely identified by using both *msgId* and *sourceNode*.

4.2 Data Structures on Nodes

Each node has a local service table which stores the information of the services provided by this node. An entry of the local service table represents a service expressed as (*id*, *address*, *functions*, *cost*). The *id* is the identification of the service, *address* is the address of this node, *functions* is the functions accomplished by this service, and *cost* is the cost of the service.

Besides, each node uses a service request table to store the SREQs that have visited this node, which is used to filter SREQ messages. Each node is aware of its own location using GPS or wireless positioning techniques [10].

4.3 Service Composition Request Transmission

Once the source node creates a SREQ and sends it towards its neighbor nodes, the SREQ will be duplicated and transmitted among nodes in the ad hoc network. The objective of each SREQ is to find a feasible service path satisfying user's distance constraint according to the function graph.

The principal processing logic of the SREQs is described as follows: When a node n_i receives a SREQ message *sreq*, it will search its local service table to discover whether there is one or more services accomplishing *nextFunction* of *sreq*. If the node cannot find any suitable service, it is considered as relay node and only adds local address to the *passedPath* of *sreq*, and transmits the message to its neighbors. If the node finds one or more feasible services, it is a service node and does the following operations:

I. If the node finds multiple feasible services, it will choose one service whose cost is minimal from the feasible services.
II. Calculates the distance from the previous service node to this node, and adds the distance to *sumDistance* of *sreq*. If there is no previous service node, the distance from the source node to the node is computed.
III. Adds the cost of the discovered service to the *sumCost* of *sreq*.
IV. Adds the local address to the *passedPath* of *sreq*.
V. Sets *preServiceNode* of *sreq* to the local node.
VI. Updates the *nextfunction* of *sreq* to the successor of this function in *functionGraph*.
VII. Finally, broadcasts the SREQ to its neighbor nodes.

For example, in Fig. 1, source node a initiates three service composition requests, i.e. $sreq1$, $sreq2$ and $sreq3$, to discover composite services accomplishing function graph $f1 \rightarrow f2 \rightarrow f3$. The $sreq1$ is sent to node b where no service $s1$ is found, so the relay node updates the *passedPath* of $sreq1$ and transmit $sreq1$ to its neighbors. After receiving $sreq1$, node c finds a service $s1$ with the function $f1$ in the local service table. Then it calculates the distance $d1$ from source node to it, and adds the distance to *sumDistance* of $sreq1$. The cost of $sreq1$ and the address of node b are added to *sumCost* and *passedPath* of $sreq1$, respectively. The *preServiceNode* and *nextFunction* of $sreq1$ is set to node b and $f2$, respectively. After updating $sreq1$, this node continues to transmit this SREQ. The $sreq1$ is processed and transmitted on other nodes in the similar way. In the end, it finds a feasible service path $\{< a,0 >,< b,0 >,< c,s1 >,< d,0 >,< e,s2 >,< f,0 >,< g,0 >,< h,s3 >\}$ whose distance is $d1 + d2 + d3$, and the cost is $c1+c2+c3$ where ci is the cost of service si.

The first problem of the above SREQ processing is that the SREQs tends to find the relatively short service paths, which may lose the chances to discover long service paths with less cost. Once discovering a service, the principal algorithm immediately searches next function of the SREQ, which do not fully use the long distance allowance. The SREQs with surplus distance should be sent to travel longer trips to find the basic

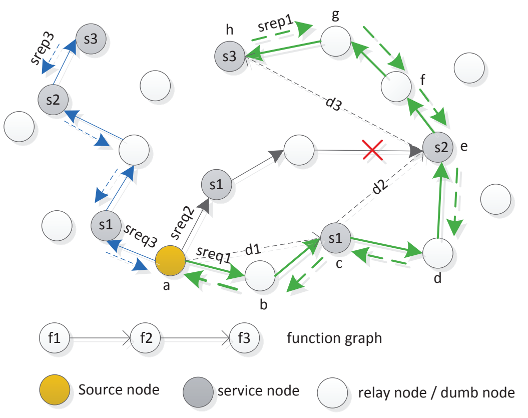


Fig. 1: Service discovery and composition

services with the less cost. So we introduce a distance vitality index to estimate the remainder distance of a SREQ after finding a feasible service path. The distance vitality index V_{req} of a SREQ message *sreq* can be calculated using the following formula.

$$V_{req} = \begin{cases} \frac{D_{max} \times N_{ds}}{D_{sum} \times N_{fun}}, & D_{sum} \neq 0 \\ \max, & D_{sum} = 0 \end{cases} \quad (4)$$

where D_{sum} and D_{max} are the *sumDistance* and *maxDistance* of *sreq*, respectively; N_{fun} is the function number in the graph function ($N_{fun} > 0$); N_{ds} is the number of the services discovered by *sreq*. $D_{sum} = 0$ means that the SREQ starts to discover the first basic service, so its vitality is maximum. $V_{req} = 1$ means the ideal case that the SREQ needs to travel the distance of D_{max} to find a service path. As an estimate metric, $V_{req} > 1$ indicates that the SREQ may have surplus distance after finding a service path, while $V_{req} < 1$ indicates that it is possibly difficult for the SREQ to find a service path within the maximum allowed distance.

When the V_{req} of a SREQ message is greater than 1, the SREQ has more distance potential to continue searching the services accomplishing the current function rather than the *nextFunction*. Otherwise, the SREQ is to immediately search the service with the function of *nextFunction*. This can lengthen the search path of the SREQ to find better service on more nodes.

The second problem induced by the principal SREQ processing is that the broadcast of SREQs may lead to broadcast storm [18], or create too much control messages in the network. So we should drop the unqualified SREQs when transmitting the messages.

To evaluate the performance of a SREQ, the goodness of a SREQ is introduced to indicate the possibility of discovering high quality service paths for the SREQ. The goodness of a SREQ message *sreq* is denoted as G_{req} which can be calculated according to the following formula.

$$G_{req} = \begin{cases} \frac{D_{max} - D_{sum}}{C_{sum}}, & C_{sum} \neq 0 \\ D_{max} - D_{sum}, & C_{sum} = 0 \end{cases} \quad (5)$$

where D_{max}, D_{sum} and C_{sum} are the *maxDistance*, *sumDistance* and *sumCost* of *sreq*, respectively. The larger is the value of SREQ's goodness, the more chance has the SREQ to find a feasible service path.

When a node receives a SREQ message $sreq_0^k$ created by a service composition request k, it searches the SREQ messages initiated by the request k in its service request table. If finding a set $S_{sreq}^k = \{sreq_1^k, sreq_2^k, \ldots, sreq_n^k\}$ of such SREQs, the goodness G_0^k of $sreq_0^k$ is compared with the goodness G_i^k of each SREQ in S_{sreq}^k. The received SREQ $sreq_0^k$ will be discarded if whose goodness satisfies the following inequality.

$$\alpha G_0^k \leq G_i^k, i = 1, 2, \ldots, n \quad (6)$$

where the weight α controls the number of the SREQs that are transmitted by this node. The larger is the weight α, the more SREQs would be transmitted. Usually, α is set to 1. In Fig. 1, for example, when the SREQ message *sreq*2 arrives at node *e*, the node compares the goodness of *sreq*2 with that of *sreq*1 that has visited the node. Because the goodness of *sreq*2 is not better than that of *sreq*1, the *sreq*2 is dropped.

Besides the goodness of SREQs, other SREQ filtering approaches are used to further reduce the SREQ messages that have less chance to find a feasible service path. They are described as follows:

(1) Transmission number filtering: From the simulations, we found that the SREQs received early by a node have more chances to get a feasible composite service than those received lately by the node. So we introduce a max SREQ transmission number (N_{max}, for short) which means a node transmits the same SREQs for at most N_{max} times. The SREQs with the identical *sourceNode* and *msgId* are considered as the same messages. The greater is the max SREQ transmission number, the more SREQs are transmitted in the network. An appropriate max SREQ transmission number can filter and drop a large number of unqualified SREQ messages.

(2) Performance filtering: For two SREQs, say *sreq*1 and *sreq*2, if *sreq*1 traverses equal or more service nodes than *sreq*2, and the former has shorter distance and spends less cost, we consider that *sreq*1 outperforms *sreq*2 in terms of distance and cost. Otherwise, we cannot certainly decide which one is better. When a node receives a SREQ message $sreq_0^k$, the message should be compared with each SREQ $sreq_i^k$ in its SREQ table. If it is not better than any $sreq_i^k$ in terms of the performances of distance and cost, it will be

discarded since a better SREQ has been transmitted by this node before. Otherwise, the $sreq_i^k$ is transmitted. This approach can avoid the presence of SREQ message loop during the message searching a service.

(3) Distance filtering: when the distance of a SREQ is greater than the maximum allowed distance, the SREQ is discarded.

(4) TTL filtering: TTL (Time-To-Live) refers to the maximum allowed hop count of a message which is set to constrain the transmission scope of control messages.

A node receives a SREQ message *sreq*0, and suppose its distance and its cost be D_{req0} and C_{req0}, respectively. D_{max}, TTL_{max} and N_{max} refer to maximum allowed distance, maximum allowed hop count, and maximum transmission number, respectively. The SREQ filtering algorithm is shown as follows.

Algorithm 1 SREQ filtering algorithm on one node

```
V_service^{sreq0} := An array of services discovered by sreq0
V_SREQ^{sreq0} := An array of the SREQs that are same with sreq0 in the SREQ table of the node
if D_sreq0 > D_max || TTL_sreq0 > TTL_max || N_req0 > N_max then
   discarding sreq0; return;
end if
for each SREQ sreq_i in V_SREQ^{sreq0} do
   if |V_service^{sreq0}| ≥ |V_service^{sreqi}| && D_sreq0 ≥ D_sreqi && C_sreq0 ≥ C_sreqi then
      discarding sreq0; return;
   end if
   α := the goodness weight
   if αG_sreq0 < G_sreqi then
      discarding sreq0; return;
   end if
end for
adding sreq0 into SREQ table of the node updating the header information of sreq0
transmitting sreq0 to the neighbors
```

4.4 Service Composition Reply

When a SREQ finds the last basic service on a node without violating the distance limitation, it has discovered a feasible service path. The node creates a Service Composition Reply (SREP) message which is denoted as $< id, sreqId, replyPath, distance, cost >$ where *id* is the identification of the SREP, *sreqId* is the *msgId* of the corresponding SREQ, *replyPath* is the reverse one of the service path discovered by the SREQ, *distance* and *cost* are the total distance and total cost of the service path, respectively. Then the last service node sends the SREP to the source node along the *replyPath* in a unicast way. For example, in Fig. 1, node *h* creates a SREP message *srep*1 according to *sreq*1, and unicasts it to source node *a* along the path $< h, g, f, e, d, c, b, a >$.

4.5 Composite Service Selection

After the source node receives the first SREP message, it would wait for a specified time period. At that time, the source node usually receives multiple SREP messages, or multiple feasible service paths. The source node chooses the best service path whose cost is minimal among the service paths. Then the best feasible service path represents the composite service satisfying users' requirements of functions, location and cost. For example, in Fig1, source node *a* receives two SREPs, *srep*1 and *srep*3, and selects *srep*1 as the composite service since the cost of *srep*1 is less than that of *srep*3.

5 Performance Evaluations

5.1 Performance Metrics

Four performance metrics are considered in our simulations. The first metric is mean *success rate* which is the ratio of the total number of successful attempts to the total number of service requests. The second one is mean *distance* of the discovered service path. The third one is mean *cost* of the discovered service path which represents the price or resource consumptions. The last metric is mean *control message overhead* which measures the load of the algorithms on network resources in terms of the number of control messages. Sending a control message over a link is counted as one message.

5.2 Service Composition Algorithms

In addition to the proposed algorithm, called location based service composition algorithm (LOCCOMP, for short), there are two typical service composition algorithms, i.e. DSRSE and DISCOMP. Dsr Service Extension algorithm (DSRSE, for short) is a service composition version of Dynamic Source Routing (DSR) protocol [19]. It extends the DSR to discover and compose services instead of discovering a destination node. A client node floods a service request to search the component services of a composite service in order. Once the service request message finds a component service in a node, the node will create a new request message to discover the next service according to the function graph. Note that a node is allowed to transmit the same service request only once. The idea of this algorithm is applied in the approaches to the execution of distributed tasks in a literature [12].

Service Discovery-Composition algorithm (DISCOMP, for short) is a well-known service composition algorithm including three phases: service discovery, service composition and route discovery. First, a node periodically discovers the services on other nodes and stores their information in a service table. Second, once receiving a service composition request, the node selects some services from its local service table and composes them. In general, the algorithm selects the component services in the vicinity of client node [1, 20]. Finally, the client node originates a routing request in search of the route that connects all selected services in order.

5.3 Simulation Environments

We implemented three service composition algorithms, including DSRSE, DISCOMP and our algorithm LOCCOMP, using Jist/Swans wireless ad hoc network simulator [21]. In our simulations, 30 nodes are randomly placed within a 1000m × 1000m area and each of them has nominal transmission range of 200 meters. The simulation time is 1000s. Four functions, i.e. f_1, f_2, f_3 and f_4, are randomly composed into some linear function graphs. The maximum distance of the discovered composite services is set to 3000m. In the network, there are four types of basic services $\{s1, s2, s3, s4\}$ where $s_i \sim f_i (i = 1, 2, 3, 4)$, which are randomly distributed on nodes. The cost of each service is randomly assigned within [0, 100). The max SREQ transmission number is set to 6. In the experiments, we studied the behaviors of the algorithms in the cases of the different count of services in the network. Service count is defined as the count of a type of services. For example, a service count of 5 means that there are five s1, s2, s3 and s4 services, respectively. We vary service count from 1 to 20 by increasing 1.

5.4 Simulation Results

We use three algorithms to do extensive simulations in order to evaluate these algorithms in terms of four performance metrics, including success rate, distance, cost and control message overhead, respectively. The simulation results are plotted in the following figures.

The success rate of each algorithm improves with the increase of service count, as shown in Fig. 2. As expected, more services provide these algorithms more chances to find feasible service paths. In most of cases, DSRSE obtains higher success rate than DISCOMP. Moreover, DISCOMP cannot guarantee to get 100% success rate even if there are a large number of services in the network. Although DISCOMP can find each basic service that is the nearest to the source node, it cannot guarantee that the path connecting the services in order must have short distance. Both LOCCOMP and DSRSE exploit request messages to discover each component

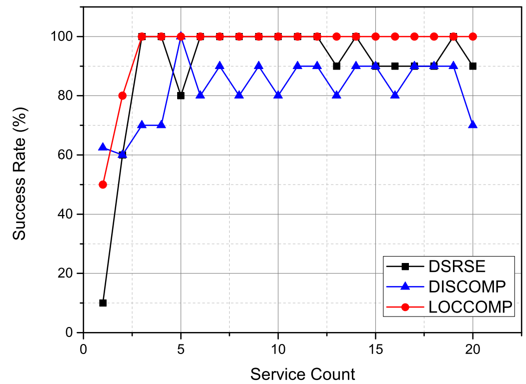


Fig. 2: Comparison of success rates

service in order, but the former obtain 96.5% success rate and the latter only obtain 89.5% in average. This is due to the fact that LOCCOMP allows the same request messages to visit a node for many times, but DSRSE does so only once. When the service count is greater than 2, our algorithm can achieve 100% success rate while other two ones cannot get such target.

The distance of the composite services discovered by these algorithms is shown in Fig. 3. With increasing the service count, the distances of three algorithms are decreased in most cases. In the three algorithms, DSRSE almost travels the longest distance to find service paths while DISCOMP travels the shortest distance. This is because DSRSE tends to discover the service nodes near the current nodes and get a long service path, but DISCOMP tries to discover the service nodes near the source node and get a relative short path in most of cases. Since our algorithm uses the distance vitality index to encourage the SREQ to travel more long path to discover more potential services, our algorithm results in a relatively long distance compared to DISCOMP when service count is less than 10. Our algorithm can obtain the comparable distance with DISCOMP in the cases of many services in the network.

Fig. 4 shows the cost of the discovered service paths. When there are few services in the network, such as $N_s < 8$ where N_s is the service count, the cost of DSRSE and DISCOMP goes up with increasing service count. The abnormal trends are induced by the fact that the small service count leads to low success rates which obviously influences the statistical results. In the cases of $N_s \geq 8$, both DSRSE and DISCOMP get relatively stable cost as well as stable success rate. Since the two algorithms merely aim to find the services with the least hops with regardless of service's cost, the increase of service number do not obviously influence the cost of composite services. Our algorithm is able to find composite services

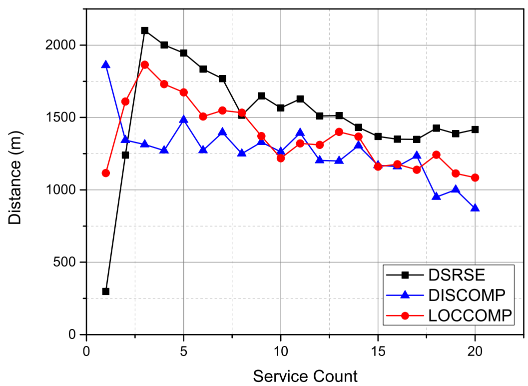


Fig. 3: Comparison of distance

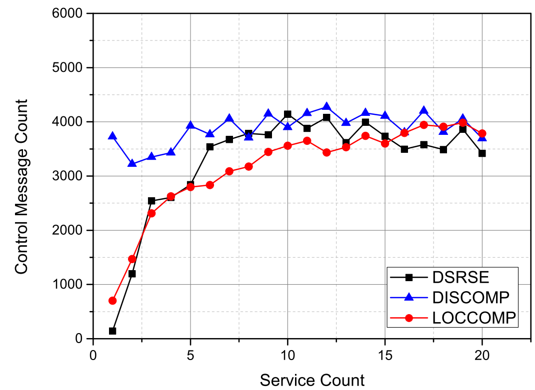


Fig. 5: Comparison of control message count

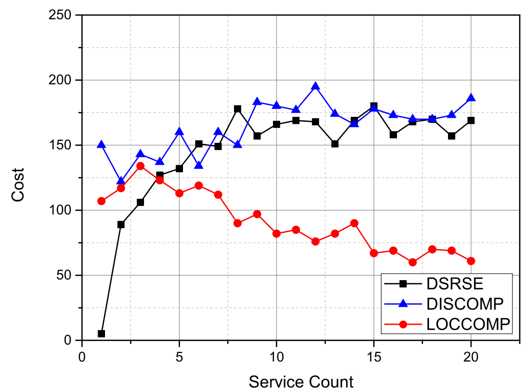


Fig. 4: Comparison of service cost

with the decrease of cost when increasing services. In most of cases, our algorithm can achieve significantly lower cost in comparison with the two other algorithms. In average, the cost of composite services found by our algorithm is only 62% of those found by DSRSE and only 56% of those found by DISCOMP, respectively.

Fig. 5 shows the control message count of these algorithms. DISCOMP almost creates more control messages than other two algorithms, and its control message count is relatively invariable whether there are few or more services in the network. This is the reason that DISCOMP broadcasts messages for two times to discover basic services and discover the routing connecting the basic services respectively, which results in a large amount of control messages in any case of service count. In the conditions of $N_s < 10$, the control messages of both DSRSE and LOCCOMP increase with increasing services, while in the conditions of $N_s \geq 10$, they keeps relatively invariable. Our algorithm produces the least control messages in the three algorithms. This is due to the fact that the effective filtering algorithm of request messages is used to drop the unqualified or surplus request messages during the message transmission.

In summary, from the above simulation results, we can see that the proposed algorithm significantly outperforms two other algorithms in terms of success rate, cost and control message count. Specifically, our algorithm can obtain higher success rate of service compositions and lower cost of composite services with using the least control messages and the comparable distance with comparison with the previous service composition algorithms.

6 Conclusions

In this paper, we present a novel distributed service composition algorithm based on location in a wireless ad hoc network. With the lack of central servers or nodes in such a network, each node employs the proposed algorithm to create, process and transmit control messages to communicate with each other. The nodes are collaborated in an effective way to search basic services, discover the routing linking the services, and compose them on the fly. In order to satisfy user's requirements of distance constraint and cost minimization, the algorithm tries to visit more nodes to find the composite services with the lower cost under the condition of distance constraint. However, this effort easily leads to more control messages which consume more resources, such as computation and memory of nodes and the bandwidth of wireless networks, and even result in network failure. Hence, to reduce control message overhead, effective control message filtering approaches, including the goodness of request messages, transmission number filtering, performance filtering, distance filtering and TTL, are applied to effectively discard the control messages with less chances to find composite services. Extensive simulation results demonstrate that the proposed algorithm can not only obtain a higher success rate and lower cost of composite services with using less control overhead when compared to other service composition algorithms.

As a future work, we will extend the proposed algorithm to achieve service compositions of more complicated function graphs which may contain conditional branches, parallel branches and hierarchical structures.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grant No. 61003022, 71001058, 71072037), Shanghai Sci. & Tech. Innovation Project (Grant No.13511505200),Humanity and Social Science Foundation of Ministry of Education of China (Grant No. 13YJA630033), and in part by Shanghai Natural Science Foundation (Grant No. 11ZR1411900, 11ZR1412100).

References

[1] D. Chakraborty, A. Joshi, T. Finin, and Y. Yesha, ”Service composition for mobile environments,” Mobile Networks and Applications, **10** , 435-451 (2005).

[2] J. Brnsted, K. M. Hansen, and M. Ingstrup, ”Service composition issues in pervasive computing,” Pervasive Computing, IEEE, **9**, 62-70 (2010).

[3] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, ”QoS-aware middleware for Web services composition,” IEEE Transactions on Software Engineering, **30**, 311-327 (2004).

[4] X. Gu, K. Nahrstedt, R. N. Chang, and C. Ward, ”QoS-assured service composition in managed service overlay networks,” in Proceedings of 23rd International Conference on Distributed Computing Systems, 194-201(2003).

[5] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, ”An approach for QoS-aware service composition based on genetic algorithms,” in Proceedings of the 2005 conference on Genetic and evolutionary computation, 1069-1075(2005).

[6] F.-Y. Chen and S.-T. Yuan, ”A Contextualized Fault-tolerant Infrastructure for P2P Mobile Service Composition,” in Proc. of IEEE International Conference on Services Computing (SCC’04), 217-224(2004).

[7] J. Gaber and M. Bakhouya, ”An Affinity-driven Clustering Approach for Service Discovery and Composition for Pervasive Computing,” in Proceedings of IEEE International Conference on Pervasive Services ICPS’06, 277-280(2006).

[8] S. Ben Mokhtar, N. Georgantas, and V. Issarny, ”COCOA: COnversation-based service COmposition in pervAsive computing environments with QoS support,” Journal of Systems and Software, **80**, 1941-1955 (2007).

[9] J. Wang, ”Exploiting mobility prediction for dependable service composition in wireless mobile Ad Hoc networks,” IEEE Transactions on Services Computing, **4**, 44-55 (2011).

[10] H. Liu, H. Darabi, P. Banerjee, and J. Liu, ”Survey of wireless indoor positioning techniques and systems,” IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, **37**, 1067-1080 (2007).

[11] X. Gu and K. Nahrstedt, ”Distributed multimedia service composition with statistical QoS assurances,” IEEE Transactions on Multimedia, **8**, 141-151 (2006).

[12] P. Basu, W. Ke, and T. D. C. Little, ”Scalable Service Composition in Mobile Ad hoc Networks using Hierarchical Task Graphs,” in Proceedings of the first Annual Mediterranean Ad Hoc Networking Workshop (2002).

[13] E. Park and H. Shin, ”Reconfigurable service composition and categorization for power-aware mobile computing,” IEEE Transactions on Parallel and Distributed Systems, **19**, 1553-1564 (2008).

[14] 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, **15**, 488-501 (2010).

[15] N. Kumar, R. Iqbal, N. Chilamkurti, and A. James, ”An ant based multi constraints QoS aware service selection algorithm in Wireless Mesh Networks,” Simulation Modelling Practice and Theory, **19**, 1933-1945 (2011).

[16] Y.-s. Luo, K. Yang, Q. Tang, J. Zhang, and B. Xiong, ”A multi-criteria network-aware service composition algorithm in wireless environments,” Computer Communications, **35**, 1882-1892 (2012).

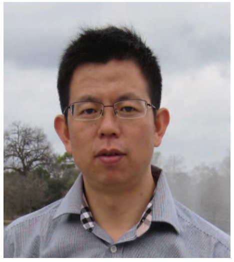
[17] S. Jiang, Y. Xue, and D. C. Schmidt, ”Minimum disruption service composition and recovery in mobile ad hoc networks,” Computer networks, **53**, 1649-1665 (2009).

[18] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, ”The broadcast storm problem in a mobile ad hoc network,” in Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, 151-162(1999).

[19] D. B. Johnson and D. A. Maltz, Dynamic source routing in ad hoc wireless networks,Kluwer Academic Publ. (1996).

[20] A. Varshavsky, B. Reid, and E. de Lara, ”A cross-layer approach to service discovery and selection in MANETs,” in Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, 1-8(2005).

[21] Cornell University. Jist/Swans. http://jist.ece.cornell.edu/ [Online] (2013).



Songqiao Han received the Ph.D degree in computer science from Shanghai Jiao Tong University in 2008. He received B.S. degree and M.S. degree from Jilin University in 1998 and 2001, respectively. He is an associate professor in Department of Computer Science and Technology at Shanghai University of Finance and Economics, China. He works as a Visiting Scholar at Northwestern University from 2013 to 2014. He has published over 20 scientific papers. His research interests are in the areas of mobile and wireless ad hoc networks, data mining , ubiquitous computing.

Xiaoling Hao received Ph.D degree in management science from Tongji University in 2004. She is currently an associate professor in School of Information Management and Engineering, Shanghai University of Finance and Economics. She worked as a Visiting Scholar at Washington University from 2009 to 2010. She is author or co-author of more than twenty scientific papers. Her research interests include financial information engineering, Operational risk management, data mining, etc.

Hailiang Huang is a professor in School of Information Management and Engineering, Shanghai University of Finance and Economics. He received his B.E. and Ph.D. in Engineering from Xi'an Jiaotong University, China in 1997 and 2002 respectively. His current research focuses on data mining and e-commerce. His research appears in a couple of international journals such as Decision Support Systems, Computers in Industry, etc.

Songxin Wang received Ph.D degree in Computer Science from Jilin University in 2002. He is currently an associate professor in Department of Computer Science and Technology, Shanghai University of Finance and Economics. His research interests include semantic web, knowledge mining and artificial intelligence.

Yong Zhang is a faculty member in the Shanghai University of Finance and Economics. He received the Ph.D. degree in computer science from Shanghai Jiao Tong University in 2009. His research interests include service selection, service configuration in mobile and pervasive computing, context-aware computing and mobile social network, etc.