PAPER

A Local Resource Sharing Platform in Mobile Cloud Computing*

Wei LIU[†a)], ***Student Member***, **Ryoichi SHINKUMA**[†], ***Senior Member***, ***and*** **Tatsuro TAKAHASHI**[†], ***Fellow***

SUMMARY Despite the increasing use of mobile computing, exploiting its full potential is difficult due to its inherent characteristics such as error-prone transmission channels, diverse node capabilities, frequent disconnections and mobility. Mobile Cloud Computing (MCC) is a paradigm that is aimed at overcoming previous problems through integrating mobile devices with cloud computing. Mobile devices, in the traditional client-server architecture of MCC, offload their tasks to the cloud to utilize the computation and storage resources of data centers. However, along with the development of hardware and software technologies in mobile devices, researchers have begun to take into consideration local resource sharing among mobile devices themselves. This is defined as the cooperation based architecture of MCC. Analogous to the conventional terminology, the resource platforms that are comprised of surrounding surrogate mobile devices are called local resource clouds. Some researchers have recently verified the feasibility and benefits of this strategy. However, existing work has neglected an important issue with this approach, i.e., how to construct local resource clouds in dynamic mobile wireless networks. This paper presents the concept and design of a local resource cloud that is both energy and time efficient. Along with theoretical models and formal definitions of problems, an efficient heuristic algorithm with low computational complexity is also presented. The results from simulations demonstrate the effectiveness of the proposed models and method.

key words: *local resource sharing, energy-efficient, mobile cloud computing*

1. Introduction

Mobile cloud computing (MCC) that is aimed at integrating mobile devices with cloud computing has been introduced in the last few years. MCC can be roughly divided into two different architectures that are client-server based and cooperation based [1]. In the traditional client-server based architecture, the cloud (data center) provides overall resource management for mobile devices. Mobile devices utilize resources in the cloud to enhance their functionality and improve their processing capabilities.

However, along with the development of hardware and software technologies, modern mobile devices (e.g., smart phones, wearable devices, and smart vehicles) have many more resources than previously, e.g., communication, computational, and information resources. These resources are not always fully utilized by their owners. Consequently, two shortcomings of the client-server based architecture have emerged: (1) Idle resources in the mobile devices themselves are not efficiently utilized. (2) Persistent connectivity to the cloud may be unavailable for mobile devices. The cooperation based architecture of MCC considers mobile devices to be part of the cloud to solve these problems. Thus, apart from utilizing resources in the traditional cloud (data center), mobile devices in the local vicinity pool and share idle resources among themselves. This approach not only makes use of pervasive resources but also enables resource sharing even when mobile devices are not able to be connected to remote clouds. The resource platform that is comprised of surrounding surrogate devices is called a local resource cloud (LRC) in this paper to differentiate it from the traditional cloud. Due to its huge potential value, the cooperation based architecture of MCC has attracted increasingly more attention from both academic and industrial communities [1]–[5]. This architecture is also called "Fog Computing" [3], "wireless distributed computing" [4] and the "vehicular cloud" in vehicular ad-hoc networks (VANETs) [5].

Error-prone transmission channels, diverse node capabilities, the mobility of nodes, and limited apriori knowledge of environments have a significant impact on the availability and reliability of devices running in mobile wireless networks. Therefore, compared with wired networks, we argue that four additional properties need to be considered when constructing LRCs in mobile wireless networks: (1) Multiple resource providers (RPs) are needed. Although resources in mobile devices have increased rapidly, they are still limited compared to traditional computing devices (e.g., PCs, servers, and clusters). As a result, resources from multiple RPs need to be gathered to facilitate task processing. (2) Physical network conditions should be integrated. Error-prone channels, locations and mobility of nodes significantly impact the quality of resource sharing. (3) The computational effectiveness of algorithms is important. Mobile wireless networks are highly dynamic due to the mobility of nodes. The LRCs must be in "real-time" and established to utilize available resources opportunistically. (4) Energy consumption should also be taken into account. Increasingly more "heavy applications" (e.g., image processing for video games, natural language processing, and streaming media) are currently favored by mobile users. These applications are expensive in terms of energy consumption. Considering the trends in mobile devices and batteries (or gasoline prices for vehicles), bottlenecks in energy consumption by mobile devices are unlikely to be solved in the near future.

However, previous work has only concentrated on veri-

Manuscript received January 20, 2014.
Manuscript revised April 30, 2014.

[†]The authors are with the Graduate School of Informatics, Kyoto University, Kyoto-shi, 606-8501 Japan.

*Part of the content of this paper has been published in proceedings of APNOMS 2013.

a) E-mail: liu@cube.kuee.kyoto-u.ac.jp

DOI: 10.1587/transcom.E97.B.1865

fying the effectiveness of the cooperative MCC to the best of our knowledge [6], [7]. Neither of them have referred to how LRCs are constructed, especially for the ones comprised of multiple RPs in dynamic mobile environments. Therefore, this paper presents the design of an LRC that selects multiple RPs to construct local resource platforms in MCC. The three main contributions of this paper are: (1) It presents the concept and design of an energy-efficient LRC in mobile networks that accelerates task completion through resource sharing among mobile devices in the local vicinity. (2) Theoretical models and a formal problem definition of an energy-efficient LRC are proposed. (3) A heuristic algorithm is presented that we prove to be efficient through extensive simulations.

The remainder of this paper is organized as follows. Based on a literature review in Sect. 2, Sect. 3 defines the concept of an LRC. Our analysis of the LRC and formal problem definitions are presented in Sect. 4. The heuristic algorithm is introduced in Sect. 5. Section 6 presents the simulation results. Section 7 concludes the paper.

2. Related Work

2.1 Mobile Cloud Computing

Much research has recently focused on the cooperative architecture of MCC. Huerta-Canepa and Lee [6] and Fitzek et al. [7] demonstrated the feasibility of cooperative resource sharing by surrounding surrogate mobile devices through experiments. The former researchers [6] presented the motivation and preliminary design for a framework to create virtual ad hoc cloud computing providers. They established prototype system based on mobile phones. Their preliminary results indicated the time efficiency of cooperation by mobile devices. The later researchers [7] considered an approach in which a centralized network (cellular) dynamically and collaboratively interacted with a local distributed network connected over short-range links (ad hoc and peer-to-peer) that aimed at achieving better use of resources (mostly energy and spectra). However, neither of them referred to how local resource platforms were to be constructed, especially in dynamic mobile environments.

Because battery capacity can't cope with the development of mobile applications, many researchers have proposed different energy-efficient solutions in the background of MCC. Cuervo et al. [8] implemented an mechanism that dynamically offloaded applications from mobile phones to surrounding infrastructures to conserve energy. Chun et al. [9] adopted VM migration to offload part of their application workload to a server with resources. Fernando et al. [2] surveyed related work in MCC.

2.2 Conventional Service Selection Algorithm

There has been much research on the selection of RPs in wired networks mainly in the field of Web services [10]–[13]. Selecting RPs that are constrained by multiple requirements is usually an NP-hard problem [14], [15]. Many heuristic approaches have been proposed to strike a tradeoff between the level of computational complexity and optimization. However, the network considered by [10]–[12] was a wired and fix network that is assumed to have constant routes and round trip time. Although Gu et al. [13] explicitly modelled link delay and availability in large wired networks, the optimized Dijkstra algorithm provided by them required a global view of the whole network. It is obvious that the assumption of constant routes or global view of the whole network is hardly holds in mobile wireless networks.

Yang et al. [14] and Luo et al. [16] took into account the selection of RPs in mobile wireless environments to satisfy quality of service (QoS) requirements. The former researchers [14] modelled availability (stability) of links based on the mobility of end nodes and took it into consideration when making decisions. The later researchers [16] proposed a heuristic method based on the Cross Entropy (CE) algorithm that preferred RPs moving slowly under the same conditions. Formal analysis of the impact of mobility was left for future work in [16]. Compared with our proposal in this paper, they focused on selecting of a single RP of best quality while neglecting the possibility of utilizing resources in multiple RPs to improve the quality of resource sharing.

3. Definitions of the Local Resource Cloud

3.1 System Model

We should take into account three steps to achieve resource sharing by mobile devices: (1) Authentication of mobile devices. Not every user of mobile devices wants to share his resources with unrelated people for security and economical reasons. However, users may be motivated to share resources with acquaintances or people who share similar interests with them [2]. As a result, the devices that take part in resource sharing should be authenticated in advance. (2) Resource discovery. Resource discovery is aimed at finding potential candidates for resource providers (RP) that own idle resources. Routes to access these candidates should also be discovered in this phase. (3) Selection of RPs. The resource requester (RR) selects a group of RPs from candidates and utilizes their resources to process its task. Since selection of RPs is independent of different authentication and resource discovery protocols, we focus on the selection phase in this paper. Cho et al. [17] and Ververidis et al. [18] surveyd related work on trust management and resource discovery protocols in mobile networks. The terminologies and abbreviations that are used throughout this paper have been summarized in Table 1.

Here, we have assumed all mobile nodes have some kind of short-range communication capabilities (e.g., WLAN ad hoc, or Bluetooth). After the resource discovery phase, the RR is aware of all the surrounding mobile nodes that have idle resources that are required (candidates of RPs) and routes to access these nodes. Multi-hop routes in mo-

Table 1 Terminologies & abbreviations.

LRC	Local resource cloud that is a resource platform composed of local mobile devices through short-range communications.
RR	A resource requester that generates an LRC to accelerate processing of its task.
RP	A resource provider that provides resources to an RR in an LRC.
RN	A relay node that contributes to the LRC by relayng packets between the RR and RP.
Task	A job generated by mobile users that consumes resources to finish it, e.g., data downloading (communication bandwidth resources) and image processing (CPU's computational resources).

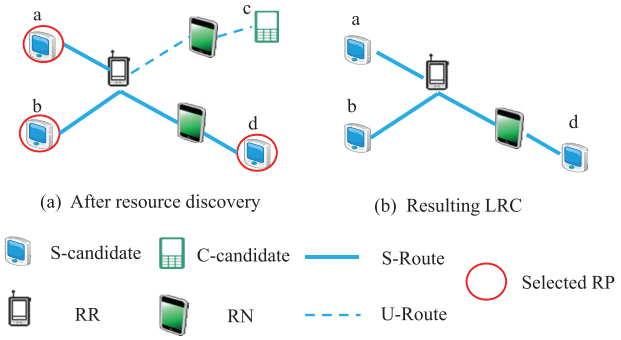

Fig. 1 System architecture.

bile networks without any infrastructure are also composed of mobile nodes. These nodes are called relay nodes (RN) in this paper. Then, the RR selects a group of RPs from all candidates to utilize their resources. As a result, the RR, selected RPs, and RNs along the routes from RR to RPs constitute a resource sharing platform through short-range wireless communications. As previously described, we called it an LRC since all the participating nodes are within the local area of RR.

Figure 1(a) shows an example scenario that four candidates of RPs (*a*, *b*, *c*, and *d*) were discovered after the resource discovery phase. Without loss of generality, candidates *a*, *b*, and *d* are assumed to be super candidates (S-candidates) that have plenty of idle resources and candidate *c* is assumed to be a common candidate (C-candidate) that has few idle resources. The routes between RR and *a*, *b*, *d* are relatively stable (S-Routes), while that between RR and *c* are unstable (U-Route)[†]. Intuitively, the RR prefers S-candidates that are connected by S-Routes for the sake of more available resources and less maintenance cost. Therefore, the RR prefers *a*, *b*, *d* over *c* in this scenario. Figure 1(b) indicates the resulting LRC if the RR selected candidates *a*, *b*, and *d* as its RPs. Theoretical models that are applicable to more complex scenarios are formalized in Sect. 4.

Two characteristics of an LRC need to be noted: (1) On-demand creation. The LRC is only created when an RR wants to utilize more resources than it owns to accelerate processing of its task. (2) Limited lifetime. After a generated task has finished, the RR releases the generated LRC to make resources in RPs available to other nodes. It is possible for multiple LRCs to simultaneously coexist in the physical network.

[†]It depends on the characteristics of mobile nodes that constitute the route, e.g., moving speed and wireless transmission range.

3.2 Problem Statement

We mainly considered the sharing of computational and communication resources among mobile devices in this paper[††]. The concept of LRCs efficiently utilizes distributed resources in surrounding mobile devices to accelerate task completion. Compared with existing solutions that only take into consideration a single RP, the LRC proposed in this paper is able to utilize more resources to achieve superior time efficiency of task processing since it comprises multiple RPs. However, extra energy is consumed by LRCs in the process of resource sharing compared with conventional computing strategies in which a user processes a generated task by itself only.

One of such consumption is the energy consumed by maintaining the structure of an LRC. The RR has to keep connected to RPs in the LRC during the process of resource sharing to utilize resources in them (e.g., if an RR that lacks the capability of Internet access wants to utilize 3G or LTE resources in other nodes, it has to send (receive) data packages to (from) RPs that own these resources through the LRC. An RR also has to coordinate and synchronize computation processes among different RPs through the LRC to share computational resources [4]). However, because nodes are mobile, routes composed of RNs are dynamic. When a route in the LRC becomes unavailable, an alternative route has to be discovered by routing protocols to guarantee communications between RR and the lost RP. Energy is consumed in discovering an alternative route. Another kind of extra energy consumption in the LRC is the energy consumed by RNs to relay packets between RR and RPs. It increases along with the increasing number of RNs (hops) along the routes from RR to RPs. It is important to minimize these two kinds of extra energy consumption while retaining time efficiency benefitted from resource sharing when constructing an LRC.

We mainly aimed at reducing the energy consumed by maintaining the structure of an LRC to improve its energy efficiency in this paper. To achieve this objective, the proposed solution prefers to select RPs connected by stable routes rather than select those connected by unstable routes. What is more, since a route with more RNs (more hops) tends to be more dynamic due to the mobility of nodes, the proposed solution still prefers to select RPs that are connected by routes with fewer RNs under the same condi-

[††]Information resources, as described by Nishio et al. [19], can be regarded as an alternative of communication and computational resources, e.g., if a Web page is cached by a node, the node does not need to consume bandwidth to receive it from Web servers.

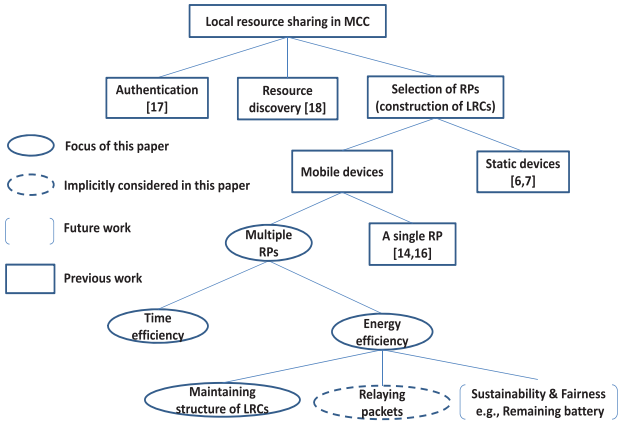

Fig. 2 Research focus of this paper.

tions. This implicitly reduces the amount of energy consumed by relaying packets in the resulting LRC. Unlike wireless sensor networks (WSNs), the lifetime of an LRC is much shorter and the depleted battery can be replaced by users in good time (e.g., at home or office). RPs and RNs can also decide whether to join an LRC according to their own statuses. As a result, we leave sustainability and fairness of LRCs (e.g., remaining battery power of RPs and RNs) for future work. Figure 2 indicates the main research focus of this paper as well as its relationship with previous work.

4. Proposed Model of the Energy-Efficient LRC

Theoretical models of an energy-efficient LRC are established in this section. Two metrics used in the models are introduced in the first two subsections: first, we define availability, which represents the dynamics of an LRC and determines the energy efficiency of an LRC; then, we define task latency, which measures the time spent in processing a task and illustrates the time efficiency of an LRC. Finally, the problem of constructing an energy-efficient LRC while retaining its time efficiency is formalized based on these two metrics.

4.1 Availability of LRC

As described in Sect. 3, energy consumed by maintaining an LRC is greatly related to the dynamics of the network. The availability of a link in mobile networks is greatly related to the mobility of its end nodes. If one node i moves beyond the transmission range of its neighboring node j, the link between nodes i and j becomes unavailable. The availability of a link also expresses the availability of a route. Consequently, the availability of an LRC depends on the availabilities of links and routes in it.

4.1.1 Availability of Links

The definition for the availabilty of a link in this paper is the same as it was in Yang et al. [14]. We briefly introduce it here for completeness and convenience. N_i and N_j are two

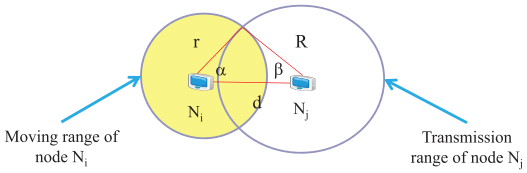

Fig. 3 Availability of links.

end nodes of a link. The transmission range of a node is R[†]. Every node moves randomly and its moving range is a circle with radius r. d is the distance between the two nodes. We assume that transmission range R is known and every node knows its location coordinates (e.g., through GPS or peer based techniques [20]). As a result, distance d between two nodes can be calculated with the Euclidean distance formula:

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \tag{1}$$

where (x_i, y_i) and (x_j, y_j) are the coordinates of nodes N_i and N_j. Finally, radius r of the moving range for a node during the process of resource sharing is its moving speed v multiplied by the period t in maintaining an LRC. Speed v of a mobile node can be calculated based on its moving distance during a period from t_1 to t_2 [21]:

$$v = \sqrt{(x_{t_1} - x_{t_2})^2 + (y_{t_1} - y_{t_2})^2}/(t_2 - t_1), \tag{2}$$

where (x_{t_1}, y_{t_1}) and (x_{t_2}, y_{t_2}) are coordinates of a node at time t_1 and t_2. The duration of period from t_1 to t_2 represents the frequency of calculating speed. Therefore,

$$r = v \times t. \tag{3}$$

As Fig. 3 indicates, the possibility of node N_i staying within the communication range of node N_j is equal to the area of node N_i's moving range inside the transmission range of node N_j divided by the overall area of node N_i's moving range. Consequently, the availability of a link composed of two nodes is:

$$A_l = \frac{Area_{inter}}{Area_{mov}}, \tag{4}$$

where $Area_{inter}$ is the intersecting area of two circles and $Area_{mov}$ is the area of N_i's moving range. Equation (4) is calculated by:

$$\alpha = \arccos\left(\frac{r^2 + d^2 - R^2}{2 \times r \times d}\right), \tag{5}$$

$$\beta = \arccos\left(\frac{R^2 + d^2 - r^2}{2 \times R \times d}\right), \tag{6}$$

$$A_l = \frac{\beta R^2 + \alpha r^2 - (R^2 \sin\beta\cos\beta) + r^2 \sin\alpha\cos\alpha)}{\pi \times v^2 \times t^2}. \tag{7}$$

[†]Although bilateral communications were assumed in this paper, this model can be applied to scenarios in which nodes have different transmission ranges.

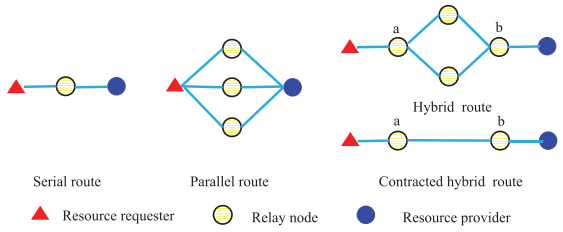

Fig. 4 Availability of routes.

The proofs of Eqs. (5), (6), and (7) are in Yang et al. [14].

4.1.2 Availability of Routes

Routes in an LRC can be divided into three subcategories of serial, parallel, and hybrid routes. Figure 4 outlines the different subcategories of routes.

a) Serial routes

All the links in serial routes are connected in sequence. A serial route is available if and only if all links in the route are available. Therefore, the availability of a serial route is:

$$A_r = \prod_{i=1}^{k} A_{l-i}, \tag{8}$$

where k is the number of links in the route and A_{l-i} is the availability of the i-th link in the route.

b) Parallel routes

There are multiple links between two end nodes in parallel routes. A parallel route is available if and only if one of the links in the routes is available. The availability of a parallel route is:

$$A_r = 1 - \prod_{i=1}^{k} (1 - A_{l-i}), \tag{9}$$

where k is the number of links in the route and A_{l-i} is the availability of the i-th link in the route.

c) Hybrid routes

Both serial and parallel routes coexist in hybrid routes. To calculate the availability of a hybrid route, we first contract all the parallel routes to a serial link as Fig. 4 indicates. The availability of the contracted link is calculated with Eq. (9). After that, the hybrid route is reduced to a serial route. Its availability can be calculated with Eq. (8).

4.1.3 Availability of LRC

The LRC generated by an RR is composed of routes to all selected RPs. As a result, the availability of an LRC is:

$$A_o = \prod_{i=1}^{q} A_{r-i}. \tag{10}$$

where q is the number of routes in the LRC and A_{r-i} is the availability of the i-th route.

4.2 Task Latency

As presented in the last subsection, period t in maintaining an LRC is a key variable in its availability. If t increases, the availability of an LRC decreases. Therefore, more energy would be consumed to maintain the LRC. According to the discussion in Sect. 3, an LRC is constructed when a task is generated by an RR and is released after the task is finished. As a result, we formally name the period t in maintaining an LRC as task latency since it illustrates the effectiveness of task processing from a user's perspective. Task latency includes three parts: the time to process the task, the transmission delay of an LRC, and the delay in maintaining routes[†].

4.2.1 Time for Task Processing

Intuitively, the time consumed to process a task decreases if the utilized resources increase. It could be simplified to the following equation:

$$T_{proc} = \frac{W}{R}, \tag{11}$$

for computational and communication resources, where W is the size of the task and R is the number of resources. For example, when using a link with throughput R bps to transmit a package with W bits, the processing time can be calculated with Eq. (11). The processing time can still be calculated with Eq. (11) for computational resources, when using CPUs that perform R operations per second to process a computing task with W operations.

4.2.2 Transmission Delay of LRC

The transmission delay of an LRC also contributes to task latency. To utilize computational and communication resources in RPs, the RR distributes subtasks to different RPs through different routes at the beginning. After the subtasks have finished, RPs return results to the RR. Consequently, a period is spent on the round trip time (RTT) for each route. Obviously, the transmission delay of an LRC is the maximum RTT value for routes in it:

$$D_{LRC} = Max_{i=1}^{q} RTT_i, \tag{12}$$

where q is the number of routes in the LRC and RTT_i is the RTT value for the i-th route. The transmission delay of an LRC can be measured in the resource discovery phase.

4.2.3 Delay in Maintaining Routes

Task latency is also influenced by the time spent to rediscover a route from the RR to a lost RP when it became

[†]In this paper, we assumed an idealized situation: the RR partitioned and distributed subtasks to RPs seamlessly. Communications between nodes through the LRC were parallel to their computational process.

unaccessible. Since the length of delay depends on different routing protocols, this variable uses a history-based method for calculation, viz., the average value of past statistics:

$$D_{route} = \frac{\sum_{i=1}^{n} D_{route-i}}{n}, \tag{13}$$

where $D_{route-i}$ is the value of the i-th record of statistics and n is the number of previous records. As a result, task latency of a generated task is:

$$t = T_{proc} + D_{LRC} + D_{route}. \tag{14}$$

D_{LRC} and D_{route} (tens or hundreds of milliseconds) are generally much shorter than T_{proc} (tens of seconds or minutes). In that case, T_{proc} approximately presents task latency.

4.3 Problem Definition of Energy-Efficient LRC

An RR generally wants to utilize resources in RPs to reduce task latency in its generated task [19]. The RR should select more RPs in the LRC to obtain more resources to achieve this goal. As the task latency decreases, the lifetime of the LRC also decreases. Consequently, energy consumed to maintain the LRC during its lifetime may decrease. However, if more RPs are selected, more routes to access these RPs need to be maintained in the same time. Therefore, energy consumed to maintain the LRC may also increase.

We assume N potential candidates of RP are discovered after the resource discovery phase and the i-th candidate has r_i units of the required resource. We define x_i as an indicator to show whether the i-th candidate is selected by the RR to be part of the LRC:

$$x_i = \begin{cases} 1 & \text{the } i\text{-th candidate is selected} \\ 0 & \text{the } i\text{-th candidate is not selected.} \end{cases} \tag{15}$$

As a result, every possible choice of an LRC is represented by an N dimentional vector X comprised of x_i. Then, the task latency of each choice is:

$$\begin{aligned} t &= T_{proc} + D_{LRC} + D_{route} \\ &= \frac{W}{\sum_{i=1}^{N} x_i \times r_i} + D_{LRC} + D_{route}. \end{aligned} \tag{16}$$

The availability of every possible choice of LRC can be calculated by substituting Eq. (16) into Eqs. (4), (8), (9), and (10). Therefore, the problem of selecting RPs to comprise an energy-efficient LRC is defined as: establish an LRC with maximum availability from all possible choices with a constraint that resulted LRC having enough resources to finish the generated task within a time threshold, T_{thresh}.

Objective: Maximize availability of the established LRC.
Constraints: $t \le T_{thresh}$, $x_i = 0$ or 1 .

This is a non-linear 0-1 programming problem that is also NP-hard (if the optimal number of RPs that need to be selected is known in advance, this problem is reduced to an NP-hard QoS-aware service composition problem [10], [15]). A greedy heuristic algorithm is introduced in the next section to solve it.

GELRC algorithm (C)

```
S = ∅ ;
C = all candidates ;
A_{o-S} = 0;

while ( the constraint of task latency is not satisfied )
   if( C == ∅ )
      task fails due to lack of resources ;
      return ;
   select i-th candidate that maximize A_{o-S} from C;
   add i-th candidate to S and delete it from C ;

S' = S ;     //S' is a temporary variable.

while ( A_{o-S} ≤ A_{o-S'} )
   S = S' ;
   select i-th candidate that maximize A_{o-S'} from C;
   add i-th candidate to S' and delete it from C ;

//make choice
S denotes resulting LRC ;
```

5. Proposed Heuristic Algorithm

As presented in the previous section, if N potential candidates were discovered in the resource discovery phase, there are 2^N different combinations of RPs. It is not feasible to compare all available choices when N is large. A greedy heuristic algorithm is proposed in this section to solve this problem. It is named as GELRC, short for Greedy algorithm for constructing Energy-efficient Local Resource Clouds.

Set C contains all discovered candidates of RP. The RR maintains set S that includes selected RPs in the LRC. It is initialized by an empty set. The availability of S is represented by A_{o-S}. First, to satisfy the constraint of task latency, the RR continues to select the i-th candidate that makes A_{o-S} the largest of possible candidates in C. If the constraint of task latency could not be satisfied even if C became empty, the task fails due to a lack of resources. After the constraint is satisfied, the RR continues to select candidates from C if and only if A_{o-S} does not decrease. Finally, the LRC denoted by S is used to process the generated task.

The asymptotic computational complexity of the proposed GELRC algorithm is $O(K \times N^2)$, where K is the maximum length of routes in the hop count. An RR generally searches for local resources with a constant hop limit, K (TTL value). In that case, the computational complexity of the method is $O(N^2)$. The pseudo-codes describe the algorithm. A step-by-step example is presented in the Appendix to illustrate it.

6. Simulations

Simulation results executed on Qualnet 5.2 are presented in this section. A pure flooding method was used for resource

discovery. Nodes were distributed randomly in a rectangular area at the beginning. Every node generated a task with an equal probability. Since the proposed algorithm for constructing an LRC was not sensitive to bandwidth, IEEE 802.11b and 2 Mbps of link bandwidth were adopted to verify its performance in a constrained environment. Because only comparisons of different methods were concerned, we assumed that transmitting a byte consumed one unit of energy and receiving a byte consumed 0.6 unit of energy for energy consumed to maintain an LRC. The transmission and energy settings were the same as those in Ahmed et al. [22].

Mobile nodes in the area were divided into three categories according to their amounts of resources:

(1) Super nodes (SN): Nodes that had 70 units of resources.

(2) Common nodes (CN): Nodes that had 30 units of resources.

(3) Relay nodes (RN): Nodes that did not have any resources.

Four different methods of constructing an LRC were compared in the simulations. The first two were the proposed methods based on the energy-efficient model described in Sect. 4.

(1) GELRC: The RR solved the energy-efficient model with the proposed heuristic algorithm.

(2) Optimal: The RR solved the energy-efficient model with a brute force strategy. The upper bound of the proposed model was obtained with this method.

(3) Random: The RR selected RPs randomly until the constraint for task latency was satisfied.

(4) LOSSA: The RR selected a single RP with best availability from all candidates that had enough resources to satisfy the constraint for task latency. This was proposed by Yang et al. [14]†.

Of the four methods, the random method was unaware of the availability of the resulting LRC. The LOSSA method only concentrated on a single SN node with best availability since only SN nodes were able to independently finish a task within the threshold of task latency. Compared with the previous two methods, both optimal and GELRC methods were able to utilize resources in multiple SN and CN nodes to increase the availability of the resulting LRC as well as reducing task latencies. The parameters used in the simulations are summarized in Table 2.

Table 2 Parameters for simulations.

Simulation area	500×500 m
Size of task	600
Number of SNs	5
Number of CNs	20
Number of RNs	25
Number of resources in SNs	70
Number of resources in CNs	30
Latency threshold	10 s
Routing protocol	DSR
MAC protocol	IEEE 802.11b
Link bandwidth	2 Mbps
Transmission range of nodes	150 m
Energy of transmission	1 unit/byte
Energy of reception	0.6 unit/byte
Mobility model	Random way point
Mobility speed	1, 5, 10, 15, 20 (m/s)
Pause time	0 s
Interval of task generation	5 s
Simulation period	1000 s

Table 3 Computation cost.

GELRC	Optimal	Random	LOSSA
0.002 s	285.318 s	$\leq$ .001 s	$\leq$ 0.001 s

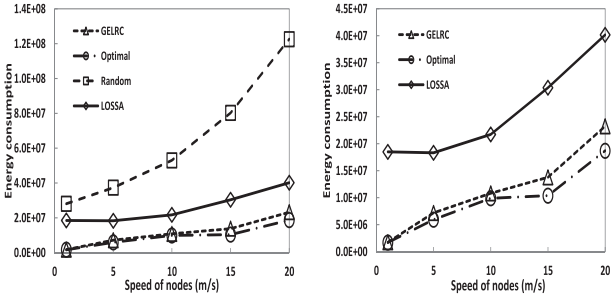

Fig. 5 Energy consumed by different methods.

6.1 Computation Cost

The first series of simulations were aimed at comparing the four methods previously described with respect to the computational overhead involved in their construction phase of an LRC. We measured the computation cost (in seconds) of selecting RPs to construct an LRC from 25 candidates with different approaches. We executed each method 10 times for each test case on a PC with an Intel Core i7-2640M CPU @ 2.80 GHz and 8.00 GB of RAM on Windows 7. The average computation costs are summarized in Table 3.

As listed in Table 3, although the optimal method achieved the upper bound for the energy-efficient model, it cost hundreds of seconds to solve it. However, nodes in realistic scenarios continue to move during this period of time. Therefore, the resulting optimal point became meaningless in the real-time distribution of nodes. The importance of the optimal method in this paper was that it specified a benchmark to measure the gap between the GELRC method and the upper bound of the proposed model.

6.2 Energy Consumption & Task Latency

The energy consumed to maintain an LRC under different speeds of nodes is plotted in Fig. 5. As Fig. 5(a) indicates, the random method consumed much more energy than the other methods. This is because the selection of RPs with the

†It was a weighted average value in Yang et al. [14]. We eliminated the "price of service" and "reliability of service" by assigning zero to their weights because these two properties were beyond the scope of this paper.

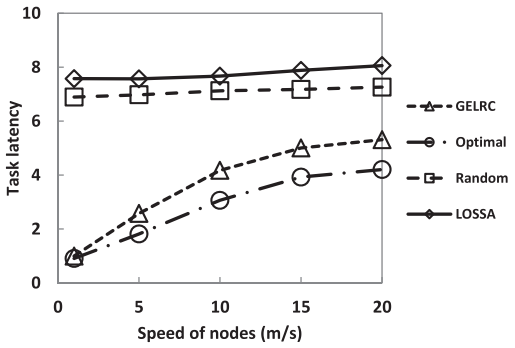

Fig. 6 Task latency of different methods.

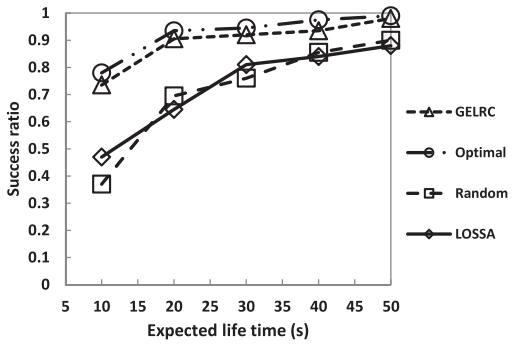

Fig. 7 Success ratios for different methods.

random method did not take into consideration the availablity of the resulting LRC. We eliminated data for the random method in Fig. 5(b) to make the comparison in other three methods clearer. As it indicates, the GELRC method consumed less energy (42–88% less according to different speeds) than the LOSSA method. This is because the GELRC method utilized available resources in multiple RPs to reduce the task latency of an LRC. The differences between the GELRC and optimal methods were small (7–36% according to different speeds).

This is also proved in Fig. 6 that plots the task latencies of the four methods. The task latencies of both the GELRC and optimal methods were much less than those of the other two methods especially when the speeds of nodes were low. The availability of an LRC remained high even if more RPs were selected when the speeds of nodes were low. Therefore, utilizing available resources in these RPs accelerated task processing while preventing from extra energy being consumed to maintain the structure of LRCs. As a result, unlike the other two methods, both methods based on the energy-efficient model were able to construct a reasonable scale LRC that was adaptive to different levels of mobility.

6.3 Failure of RPs

We assumed that all RPs in an LRC remained "on-line" from the begining to the end of task processing in previous simulations. However, in reality, mobile RPs may suddenly go "off-line" due to reasons like system crashes or moving out of communication range. The RR is usually not capable of predicting this kind of "sudden failure of RPs" or is not notified by RPs in advance. In this kind of scenario, the constraint for task latency can't be guaranteed. As a result, we defined a metric called "success ratio" to measure the capabilities of fault tolerance by different methods. In this paper, the "success ratio" is defined as the percentage of tasks that are finished within the threshold of task latency without reestablishing a new LRC. The lifetime of a node is defined as the duration over which it is capable of sharing resources with other nodes in the following descriptions.

We assumed the lifetimes of both SN and CN nodes conformed to an exponential distribution (RNs were not taken into account because they have no resources. The failure of RN nodes was the same as the failure of simulated routes discussed in the previous subsection). Nodes were disabled after their lifetimes expired. If a disabled node acted as an RP in an LRC, the RR of the LRC detected the node had failed through a keep-alive timer. The keep-alive timer was set to 2 s in the simulation[†]. After the disabled node was detected, we assumed the RR could ideally distribute remaining subtasks for the disabled node to all remaining RPs including itself (when it had resources). A new node with the same number of resources was enabled at a random location to replace the disabled node to keep node densities constant. The resulting success ratios of different methods are plotted in Fig. 7 (speed of nodes was 10 m/s). According to the results, both the GELRC and optimal methods outperformed the other two methods in dynamic environments. This mainly benefitted from the fact that both GELRC and optimal methods reduced task latencies as shown in Fig. 6. As a result, (1) the probability of the failure of RPs within a shorter period of task processing time decreased, and (2) a larger room was reserved for the failure of RPs (e.g., even if task latencies were delayed by 4 seconds due to the failure of RPs, both GELRC and optimal methods were able to finish it within the threshold (10 seconds) as shown in Fig. 6. LOSSA and random methods failed to do this).

Based on previous simulation results, it can be concluded that: (1) The proposed model and methods are able to construct an LRC that is both energy and time efficient in mobile wireless networks. (2) When the failure of nodes was taken into consideration, the proposed methods were capable of higher success ratios than the conventional methods. (3) The proposed GELRC method performed close to the upper bound (optimal method) of the energy-efficient model while dramatically decreasing the compuation cost.

7. Conclusions

The design of an energy-efficient local resource cloud (LRC) in cooperative MCC was proposed in this paper. It is constructed of an on-demand local resource platform to

[†]Two seconds was enough for this simulation because the RTTs of most routes were within 200 ms. This value could be adjusted according to different network statuses.

enable resource sharing among mobile devices. Simulation results indicated that the proposed model and methods were efficient in terms of both energy consumption and time. The proposed heuristic method (GELRC) with low computational complexity is suitable for dynamic mobile wireless environments. For future work, explicit considerations of energy consumed by relaying packets, sustainability and fairness of LRCs are interesting research areas in local resource sharing.

Acknowledgment

This work was supported in part by the National Institute of Information and Communications Technology (NICT), Japan.

References

[1] L. Guan, X. Ke, M. Song, and J. Song, "A survey of research on mobile cloud computing," Computer and Information Science (ICIS), 2011 IEEE/ACIS 10th International Conference on, pp.387–392, IEEE, 2011.
[2] N. Fernando, S.W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," Future Generation Computer Systems, vol.29, no.1, pp.84–106, 2013.
[3] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," Proc. First Edition of the MCC Workshop on Mobile Cloud Computing, pp.13–16, 2012.
[4] D. Datla, X. Chen, T. Tsou, S. Raghunandan, S.S. Hasan, J.H. Reed, C.B. Dietrich, T. Bose, B. Fette, and J. Kim, "Wireless distributed computing: A survey of research challenges," IEEE Commun. Mag., vol.50, no.1, pp.144–152, 2012.
[5] S. Olariu, I. Khalil, and M. Abuelela, "Taking vanet to the clouds," Int. J. Pervasive Computing and Communications, vol.7, no.1, pp.7–21, 2011.
[6] G. Huerta-Canepa and D. Lee, "A virtual cloud computing provider for mobile devices," Proc. 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond, p.6, 2010.
[7] F.H. Fitzek, M. Katz, and Q. Zhang, "Cellular controlled short-range communication for cooperative p2p networking," Wireless Pers. Commun., vol.48, no.1, pp.141–155, 2009.
[8] E. Cuervo, A. Balasubramanian, D.k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," Proc. 8th International Conference on Mobile Systems, Applications, and Services, pp.49–62, 2010.
[9] B.G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," Proc. Sixth Conference on Computer Systems, pp.301–314, 2011.
[10] L. Zeng, B. Benatallah, A.H.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "Qos-aware middleware for web services composition," IEEE Trans. Softw. Eng., vol.30, no.5, pp.311–327, 2004.
[11] E.M. Maximilien and M.P. Singh, "A framework and ontology for dynamic web services selection," IEEE Internet Comput., vol.8, no.5, pp.84–93, 2004.
[12] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," IEEE Trans. Softw. Eng., vol.33, no.6, pp.369–384, 2007.
[13] X. Gu, K. Nahrstedt, and B. Yu, "Spidernet: An integrated peer-to-peer service composition framework," Proc. 13th IEEE International Symposium on High performance Distributed Computing, 2004, pp.110–119, 2004.
[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, vol.15, no.4, pp.488–501, 2010.
[15] G. Canfora, M. Di Penta, R. Esposito, and M.L. Villani, "An approach for qos-aware service composition based on genetic algorithms," Proc. 2005 Conference on Genetic and Evolutionary Computation, pp.1069–1075, 2005.
[16] Y.s. Luo, K. Yang, Q. Tang, J. Zhang, and B. Xiong, "A multi-criteria network-aware service composition algorithm in wireless environments," Comput. Commun., vol.35, no.15, pp.1882–1892, 2012.
[17] J.H. Cho, A. Swami, and R. Chen, "A survey on trust management for mobile ad hoc networks," IEEE Communications Surveys & Tutorials, vol.13, no.4, pp.562–583, 2011.
[18] C. Ververidis and G. Polyzos, "Service discovery for mobile ad hoc networks: A survey of issues and techniques," IEEE Communications Surveys & Tutorials, vol.10, no.3, pp.30–45, 2008.
[19] T. Nishio, R. Shinkuma, T. Takahashi, and N.B. Mandayam, "Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud," Proc. First International Workshop on Mobile Cloud Computing & Networking, pp.19–26, 2013.
[20] R. Mayrhofer, C. Holzmann, and R. Koprivec, "Friends radar: Towards a private P2P location sharing platform," Computer Aided Systems Theory — EUROCAST 2011, pp.527–535, Springer, 2012.
[21] Y.B. Ko and N.H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," Proc. 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking, pp.66–75, 1998.
[22] A. Ahmed, K. Yasumoto, N. Shibata, and T. Kitani, "Hdar: Highly distributed adaptive service replication for manets," IEICE Trans. Inf. & Syst., vol.E94-D, no.1, pp.91–103, Jan. 2011.

Appendix

An example is presented to illustrate the GELRC algorithm proposed in Sect. 5. The scenario outlined in Fig. 1 is used in this part. At the beginning of executing the GELRC algorithm, the set S was empty, $S = \emptyset$. The set C contained all discovered candidates of RPs, $C = \{a, b, c, d\}$.

The RR firstly selected candidate a into S if the availability of S, $A_{o-\{a\}}$, was larger than $A_{o-\{b\}}$, $A_{o-\{c\}}$, and $A_{o-\{d\}}$. Then $S = \{a\}$ and $C = \{b, c, d\}$. Assuming that the task latency of resulting S was still larger than the value of threshold, the RR had to select RPs from remaining candidates to satisfy the constraint. Therefore, candidate b was selected if the availability of S, $A_{o-\{a,b\}}$, is larger than $A_{o-\{a,c\}}$ and $A_{o-\{a,d\}}$. After that $S = \{a, b\}$ and $C = \{c, d\}$. Assuming that the constraint of task latency was satisfied after selecting b into S, the RR continued to search for additional resources in other RPs to further reduce the task latency with a constraint that selecting that RP into S did not decrease the availability of the resulting LRC. It was possible because although an additional route from RR to the selected RP had to be maintained, additional resources in the selected RP also reduced the maintaining period of all the routes in the LRC. As a result, candidate d was selected into S if $A_{o-\{a,b,d\}}$ was larger than $A_{o-\{a,b,c\}}$ and $A_{o-\{a,b\}}$. Then, $S = \{a, b, d\}$ and $C = \{c\}$. At last, assuming that $A_{o-\{a,b,c,d\}}$ was less than $A_{o-\{a,b,d\}}$, the RR gave up candidate c and used an LRC comprised of $\{a, b, d\}$ to process its task.

Wei Liu received the B.E. and M.E. degree in Software Engineering from Chongqing University, China, in 2006 and 2009. He is currently working toward the Ph.D. degree in Communications and Computer Engineering, Graduate School of Informatics, Kyoto University. His current research interests include overlay networks and mobile cloud computing.

Ryoichi Shinkuma received the B.E., M.E., and Ph.D. degrees in Communications Engineering from Osaka University, Japan, in 2000, 2001, and 2003, respectively. In 2003, he joined the faculty of Communications and Computer Engineering, Graduate School of Informatics, Kyoto University, Japan, where he is currently an Associate Professor. He was a Visiting Scholar at Wireless Information Network Laboratory (WINLAB), Rutgers, the State University of New Jersey, USA, from 2008 Fall to 2009 Fall. His research interests include network design and control criteria, particularly inspired by economic and social aspects. He received the Young Researchers' Award from IEICE in 2006 and the Young Scientist Award from Ericsson Japan in 2007, respectively. He is a member of IEEE.

Tatsuro Takahashi received the B.E. and M.E. in Electrical Engineering from Kyoto University, Kyoto, Japan, in 1973 and 1975 respectively, and Dr. of Engineering in Information Science from Kyoto University in 1997. He was with NTT Laboratories from 1975 to 2000, making R&D on high speed networks and switching systems for circuit switching, packet switching, frame relaying, and ATM. Since July 1, 2000, he is a Professor, Communications and Computer Engineering, Graduate School of Informatics, Kyoto University. His current research interests include high-speed networking, photonic networks and mobile networks. Prof. Takahashi received the Achievement Award from IEICE in 1996, the Minister of Science and Technology Award in 1998, and the Distinguished Achievement of Contributions Award from IEICE in 2011. He was a Vice President of the ATM Forum from 1996 to 1997, and the Chairman of the Network Systems (NS) Technical Group in the Communications Society of IEICE from 2001 to 2002. Prof. Takahashi is an IEEE Fellow.