PAPER

# HDAR: Highly Distributed Adaptive Service Replication for MANETs

**Asaad AHMED**[†a)], ***Nonmember***, **Keiichi YASUMOTO**[†], **Minoru ITO**[†], ***Members***, **Naoki SHIBATA**[††], ***Nonmember***, ***and*** **Tomoya KITANI**[†††], ***Member***

**SUMMARY** Mobile Ad Hoc Networks (MANETs) offer quick and easy network deployment in situations where it is not possible otherwise and they can be used to provide mobile users with a temporary infrastructure to use services in the absence of fixed infrastructure. Nodes in MANETs are free to move and organize themselves in an arbitrary fashion. The challenging task in such dynamic environments is how to improve the service availability. Replicating a service at some nodes distributed across the network is an effective strategy. However, service replication can considerably impact the system energy consumption. Since mobile devices have limited battery resources, a dynamic and efficient service replication is necessary to support such environments. In this paper, we propose a distributed service replication scheme for achieving high service availability with reasonable energy consumption for MANETs. The proposed method called HDAR (Highly Distributed Adaptive Service Replication) divides the whole network into disjoint zones of at most 2-hops in diameter and builds a dynamic replication mechanism which selects new replica zones depending on their service demand and the tradeoff between the communication and replication energy consumption costs. Through simulations, we confirmed that our approach can achieve higher service availability with reasonable energy consumption than existing methods.

***key words:*** *MANET, service replication, service availability, energy consumption*

## 1. Introduction

Mobile Ad Hoc Networks (MANETs) are autonomous collections of mobile users that communicate over relatively bandwidth-constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably overtime. With the advances in wireless networking technologies and the advent of portable devices, MANET applications can offer various services and resources to users such as multimedia information service, file-sharing service, database retrieval service, location service, etc. In such applications, users need to detect, share, and invoke the services and resources in a flexible manner. To build a model for those applications, we need a way to organize and maintain service objects based on *Service Oriented Architecture* (SOA) [1].

Manuscript received March 11, 2010.
Manuscript revised August 25, 2010.

[†]The authors are with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Ikomashi, 630–0192 Japan.
[††]The author is with the Department of Information Processing and Management, Shiga University, Hikone-shi, 522–8522 Japan.
[†††]The author is with the Division of Global Research Leaders, Shizuoka University, Hamamatsu-shi, 432–8011 Japan.
a) E-mail: asaad-g@is.naist.jp

DOI: 10.1587/transinf.E94.D.91

Consider the following application scenario: pedestrians with mobile terminals in a city have some sensors. They want to share situations of the city by temperature, humidity, human density, noise, illuminance, etc. on any geographical point in the target area. This can be realized with SOA as follows: all mobile nodes sense data and send it to a service provider node. The service provider node constructs a database with the collected data and provides a retrieval service. A mobile node sends a query to the service provider node with the location where the node wants to know the situation. In this scenario, since WLAN does not cover the whole city and cellular networks may be overloaded with high density areas or results in high cost (e.g. people may not want to pay for data upload), MANETs can be used as alternative networks to realize this SOA.

The basic components to realize SOA are services (objects and providers), clients, and a service discovery mechanism. Due to dynamic nature of MANETs, a service provider may be either temporary or even permanently unavailable for the client nodes, because the provider node leaves the network and/or battery of the provider node is depleted due to communication and computation loads. From the client's point of view, a service must be available regardless of these reasons. Also, mobile devices have a limited amount of battery, so the energy consumption for retrieving service across the network has to be minimized. Service replication is an effective strategy to satisfy these goals.

There have been proposed several methods that replicate a service to some new host nodes based on particular strategies such as a whole knowledge of the network [2], [3], network partitioning [4]–[8], network density [9], or client's request rate [10]. Most of these methods focus on how to increase the service availability without taking into account the energy consumption. Thus, we need to design a dynamic replication strategy to select a limited number of nodes to act as service providers that balance energy consumption and the service availability without full knowledge of the network. In addition, the path length between a client and a server is another important metric, because if the path between the client and the server is too long, the client may not access the service and the energy consumption will increase. So, we need to minimize the path length between the client and the server. Here, we use the same definition of service availability as in [7], [10], that is, the ratio of the number of service replies received to the number of service requests sent.

In this paper, we propose a distributed replication scheme called *Highly Distributed Adaptive Service Replication (HDAR)*, aiming to improve service availability with reasonable energy consumption across the network. In order to dynamically place service replicas in appropriate nodes, HDAR divides the whole network into disjoint zones with diameters at most 2 hops, selects a node with minimum moving speed in each zone as a zone head, and constructs a virtual backbone network connecting all zone heads. To determine the place of a new service replica, HDAR builds a dynamic replication mechanism which aims to replicate a service dynamically to some of the zones depending on the service demand level in each zone and the tradeoff between the communication and replication energy consumption costs. In addition, to control the number of service replicas, HADR lets $r$-level neighboring servers exchange with each other information of their covering zones ($r$ is a control parameter which limits the number of neighboring servers).

Through simulations, we have confirmed that our approach can achieve higher service availability with reasonable energy consumption than existing methods.

The rest of this paper is organized as follows. Related work is introduced in Sect. 2. Service availability problem is formulated in Sect. 3. The proposed HDAR is described in Sect. 4. Performance evaluation is provided in Sect. 5 and Sect. 6 concludes the paper.

## 2. Related Work

There have been proposed several methods that replicate a service to some new host nodes based on particular strategies. In [7], [8], Derhab et al. proposed a method utilizing replication and merging mechanisms based on estimation of the link quality and partition prediction by using the TORA [12] and some partition detection mechanisms. REDMAN middleware [9] aimed to support resource replication in dense MANETs. By using a simple gossip-based strategy, each node randomly decides whether to host a replica on its storage and successively forwards the resources to be replicated to one of its neighbors.

In [2], [3], the replication process depends on the whole knowledge of the network where nodes should require the information about the other nodes in the network. By using link quality to predict network partitioning, the original service is replicated to the node with high battery lifetime in the partition. In [4], each client monitors the set of disjoint paths between itself and the server and computes a certain metric. If this metric falls below a certain threshold then a potential partition is identified and server replication is initiated. In [5], [6], a partition prediction model was proposed based on grouping of nodes according to their position and speed. Every client sends its coordinates and velocity to the server. Having this global knowledge, the server can predict future partitions and a new server is replicated accordingly. In [10], service distribution protocol *(SDP)* was proposed based on clients' and providers' interests. In this algorithm, there are two mechanisms: (1) service replication where the service is replicated to a new node if its interest exceeds some predefined threshold called *replication threshold*; and (2) service hibernation where the service is hibernated from the service provider if its interest is less than or equal to some predefined threshold called *hibernation threshold*.

Most of the aforementioned approaches focus mainly on service availability improvement in the case that network partitioning is predicted. When a partition is going to happen, the requested service is replicated in advance and connectivity to it can be guaranteed. The other approaches use another strategy such as gossip-based strategy [9] or fixed threshold such as client's interest [10]. However, all of them focus on how to increase the service availability without taking into account the energy consumption.

In [11], we proposed a distributed adaptive service replication (DAR) that achieves high service availability with reasonable energy consumption. DAR uses a zone-based architecture to replicate the service dynamically to some of zones depending on the service demand level in each zone. In DAR, each server executes the replication mechanism independently of other servers and does not take into account the balance among the covering zones of servers. In this paper, we propose HDAR. In DAR, the path length between a client and a server depends on the network size where the path length decreases as network size increases. In this paper, to eliminate this dependability, we propose HDAR. In HDAR, the replication mechanism depends on the service demand level in each zone as DAR does. In addition, HDAR considers the tradeoff between the communication and replication energy consumption costs as another parameter. Also, to control the number of service replicas, HADR lets $r$-level neighboring servers exchange with each other information of their covering zones.

The path length between a client and a server in most of existing methods depends on the network size. In other words, with existing methods, the path length decreases as network size increases. Unlike them, our contribution here is the proposal of a new service replication method that improves the service availability with reasonable energy consumption and minimizes the path length between a client and a server independently of network size.

## 3. Problem Formulation

In this section, we formulate the problem to improve service availability when realizing SOA in MANETs.

### 3.1 Models, Assumptions, and Definitions

*1) Network Model*: Hereafter, we use discrete time and represent the current time by positive integer variable $t$. A network at time $t$ which consists of a set of servers and a set of clients is modeled as an undirected graph $G(t) = (V(t), E(t))$, where $V(t)$ is the set of nodes at time $t$ and $E(t) \subseteq V(t) \times V(t)$ is the set of links among the nodes in $V(t)$ at time $t$. We assume that all nodes are cooperative and there is no any

selfish node in $V(t)$. We denote the set of servers and the set of clients at time $t$ as $S(t)$ and $C(t)$, respectively. Here, $V(t) = S(t) \cup C(t)$ and all nodes in $V(t)$ communicate through omni-directional antennas with some nodes in their transmission range denoted by $T_R$. $T_R$ is assumed to be a disk with a certain radius centered at the sender node. We assume that each node $v \in V(t)$ has a unique *ID* and a *Received Signal Strength Indicator* (RSSI) capability, and knows its moving speed with some means. A link $(u, v) \in E(t)$ exists iff $u$ and $v$ are within the transmission range of each other at time $t$. We assume that $G(t)$ is connected for any $t$. This means that for any pair of nodes $u$ and $v$, when $u$ and $v$ are not in the transmission range of each other, they can communicate with each other in a *multi-hop* manner through other nodes in the network. We denote the number of hops in the shortest path between any two nodes $u, v \in V(t)$ by $d(u, v, t)$. We assume that the shortest path between any two nodes is known through the lower layer routing protocol.

*2) Service Model*: We assume that each client knows the set of available servers and it can contact with a server by using closest (shortest hop) server selection scheme such as [13]. Each client sends a service request to the closest server with its *ID* and service parameters which depend on the type of application. In order to maintain loose data consistency among servers, we assume that each server periodically sends an update request including new data to other servers every long time period (e.g. every 1 hour). So, the cost of updating is not considered here. We assume that only one packet is needed to send the service request or the service reply message between the client and the server. We denote the service request path from the client $c$ to the server $s$ at time $t$ and the service reply path from the server $s$ to the client $c$ at time $t$ as $d(c, s, t)$ and $d(s, c, t)$, respectively. We assume that the energy amounts required to transmit and receive one packet along any link $(u, v) \in E(t)$ are fixed and denoted by $E_{tr}$ and $E_{rx}$, respectively. We assume that the server $s$ executes the replication process to select new service replicas every specified time interval called *replication interval* denoted by $RI$. We denote the number of service requests received from the client $c \in C(t)$ at server $s$ per unit time at time $t$ by $sq(c, s, t)$. We assume that $s$ knows the value of $sq(c, s, t)$ locally by counting the number of service request messages received from $c$ and the value of $sq(c, s, t)$ is initialized every replication interval $RI$.

*3) Replication cost*: Service availability is improved by replicating a service to a set of nodes across the network. However, replication itself imposes additional energy consumption for the nodes that transmit and receive the service object (i.e., program and data). Let $R_s(t)$ denote the set of new replica nodes which was determined by $s \in S(t)$ at time $t$. We assume that $k$ packets are used to send the service object. The amount of consumed energy for the set of new replica nodes $R_s(t)$ at time $t$ is denoted by $RepCost(s, R_s(t), t)$ and defined as:

$$RepCost(s, R_s(t), t) = k \times (E_{tr} + E_{rx}) \times \sum_{r \in R_s(t)} d(s, r, t) \quad (1)$$

*4) Communication cost*: The service request or the service reply from/to the client $c$ to/from the server $s$ is transmitted through a one-hop or multi-hop path. During transmission, each node in the path consumes energy for receiving and transmitting a message. We model a communication cost for the client $c \in C(t)$ by the energy consumption to communicate with the server $s$ along the service request path $d(c, s, t)$ and the service reply path $d(s, c, t)$. The communication cost to communicate with the server $s$ during $RI$ from time $t$ is denoted by $ComCost(s, C(t), t)$ and defined as:

$$ComCost(s, C(t), t) = (E_{tr} + E_{rx}) \times RI \times \sum_{c \in C(t)} sq(c, s, t)[d(c, s, t) + d(s, c, t)]z(c, s) \quad (2)$$

where $z(c, s)$ is an assignment such that $z(c, s) = 1$ if the client $c$ accesses the server $s$, otherwise $z(c, s) = 0$.

### 3.2 Problem Definition

Given a network $G(t)$, and the number of service requests $sq(c, s, t)$ from each client $c \in C(t)$ to each server $s \in S(t)$ at time $t$, the total amount of energy consumption for communication and replication $Cost(S(t), C(t), t)$ at time $t$ is defined as:

$$Cost(S(t), C(t), t) = \sum_{s \in S(t)} [\sum_{s' \in R_s(t)} ComCost(s', C(t), t) + RepCost(s, R_s(t), t)] \quad (3)$$

Our objective is to find a set of replica nodes $R_s(t)$ for each server $s \in S(t)$ which guarantees a certain level of service availability and minimizes the total amount of energy consumption for communication and replication $Cost(S(t), C(t), t)$. So, the objective function is defined as follows.

$$\textbf{Minimize} \quad Cost(S(t), C(t), t) \quad (4)$$

**subject to**

$$\sum_{s \in S(t)} ComCost(s, C(t), t) - \sum_{s \in S(t)} \sum_{s' \in R_s(t)} ComCost(s', C(t), t) > RepCost(s, R_s(t), t) \quad (5)$$

Constraint (5) indicates the difference between the communication cost of current set of servers and that of new set of servers (after replication) is larger than the replication cost.

## 4. Highly Distributed Adaptive Service Replication Algorithm (HDAR)

Our target problem defined in Sect. 3 is the combinatory optimization problem and is closely related to Uncapacitated Facility Location Problem (UFLP) which is known to be NP-hard [14]. In this section, we propose a heuristic but

**Table 1** HDAR and DAR comparison.

| Protocol | Zone-based architecture | Replication mechanism parameters | |
|---|---|---|---|
| | | Service demand level of a zone | Communication and replication cost tradeoff estimation |
| DAR [11] | yes | yes | no |
| HDAR | yes | yes | yes |

fully distributed service replication algorithm named HDAR to solve the problem. HDAR is based on a zone structure where the whole network is divided into disjoint zones. The zone-structure is often used to solve the scalability issue in large MANETs [15]–[17].

### 4.1 Basic Idea

Similarly to our previous method DAR [11], HDAR divides the whole network into disjoint zones. A major goal of constructing zones is to organize all zone heads into a virtual backbone network to simplify the system control, decrease message overhead, and manage the service replication mechanism. HDAR forms zones and selects zone heads according to a mobility metric because the zone-based structure requires stable zones even in the presence of node mobility. Thus, HDAR selects a node with minimum moving speed in each zone as a zone head, and constructs a virtual backbone network connecting all zone heads. By using our zone structure, HDAR provides dynamic replication that determines the number and location of new service replicas. HDAR aims to replicate a service dynamically to some of zones depending on (i) the service demand level and (ii) the tradeoff between communization and replication costs for each zone. Table 1 shows the protocol design and behavior differences between DAR and HDAR.

Our goals are as follows: (a) minimizing energy consumption depending on server workload (e.g. communication and computation loads) and communication cost between clients and the server; and (b) improving the service availability independently of network size (number of nodes). Hereafter, a zone is called *active* if it has at least one node requesting a service during the replication interval, otherwise the zone is called *passive*.

### 4.2 Zone Formation and Maintenance

In our zone structure, a node is in one of the following four states.

***Zone_Undecided***: *A node which does not belong to any zone.*

***Zone_Head***: *A node which has become a zone head.*

***Zone_Member***: *A node which belongs to a zone.*

***Zone_MemberGeteway***: *A node which is a zone member and a neighbor to at least one member of another zone.*

Now, we will describe the formation and maintenance processes in the following algorithm[†]:

First, all nodes periodically exchange *hello* message with their neighbors (in 1-hop). Each hello message contains: node id, node speed, node residual energy, node capacity, node state, and other needed parameters. Secondly, all nodes start in *Zone_Undecided* state. Finally, each node changes its state as follows: (i) if a node has the lowest speed amongst all of its neighbors, its state becomes *Zone_Head*, otherwise its state becomes *Zone_Member*, (ii) if two neighboring nodes in a *Zone_Undecided* state have the same lowest speed, the node with the lowest *ID* will be selected as a zone head and the node's state becomes *Zone_Head*, and the other node's state is changed to *Zone_Member*, (iii) if a *Zone_Member* node is a neighbor to at least one member of another zone, the node's state becomes *Zone_MemberGateway*, (iv) if two nodes with state *Zone_Head* move into each other's radio range, then the node with the lowest speed (lower ID if the speed is the same) remains in the state *Zone_Head* and the other node's state is changed to *Zone_Member*, and (v) if a *Zone_Undecided* node becomes a neighbor of more than one node with state *Zone_Head*, it uses the RSSI of hello messages received from these *Zone_Head* nodes, selects the stronger RSSI *Zone_Head* node and becomes a *Zone_Member* node in its zone.

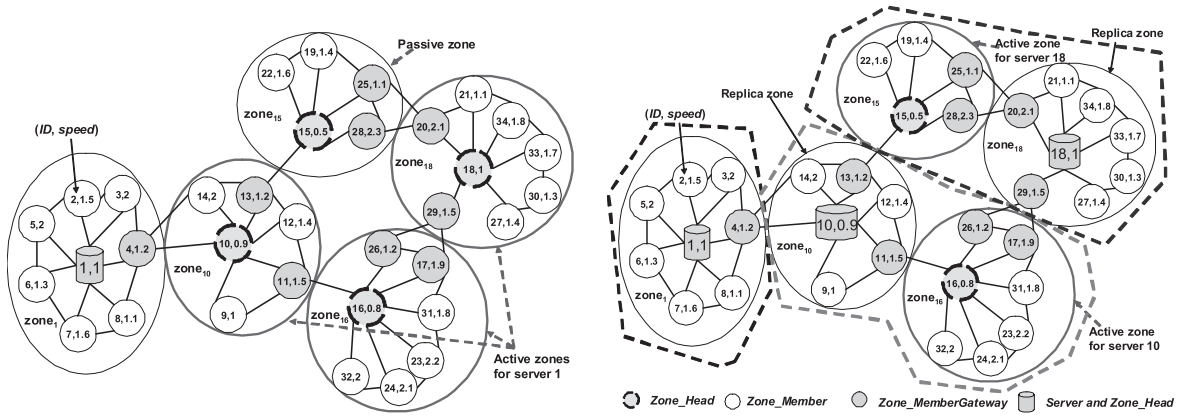
This algorithm leads to the formation of disjoint zones which are at most 2-hops in diameter. Figure 1 (a) shows the result of applying the above algorithm, where each node is represented by (*ID*, *speed*) pair.

The zone-structure of HDAR and DAR is the same as 1-hop zone-based routing protocols. Some of these routing protocols use the lowest ID [16] or the highest degree of node [17] to select a zone head. While, HDAR and DAR use the node's speed to select a zone head for each zone. Where HDAR and DAR select the node with the minimum speed among its neighbor nodes as a zone head.

### 4.3 The HDAR Replication Mechanism

In HDAR, each client asks its zone head $zh$ to forward the service request message. After that $zh$ adds to the message the number of nodes in its zone, $n_{zone_{zh}}(t)$, and forwards the message toward the nearest server. In HDAR, each server $s$ can serve multiple zones, so it maintains an *Aggregated Request Table* (*ART*) which stores the zone identifier $zone_{zh}$, the number of received service requests $sq(s, zone_{zh}, t)$, the number of nodes $n_{zone_{zh}}(t)$, and the number of hops $d(s, zone_{zh}, t)$ for every active zone $zone_{zh}$ which accesses $s$, as shown in Table 2. As described in the service model in Sect. 3, the server $s$ can know received service demand information from the received service request messages.

[†]The algorithm is designed to run continuously and asynchronously on each active node in the network.




(a) (b)

**Fig. 1** Zone formation, sever, replicas, and active zones.

**Table 2** Aggregated request table, ART.

| zone Id $zone_{zh}$ | number of received service requests $sq(s, zone_{zh}, t)$ | number of nodes $n_{zone_{zh}}(t)$ | number of hops $d(s, zone_{zh}, t)$ |
|---|---|---|---|
| $zone_{10}$ | 6 | 6 | 2 |
| $zone_{16}$ | 3 | 7 | 4 |
| $zone_{18}$ | 10 | 8 | 7 |

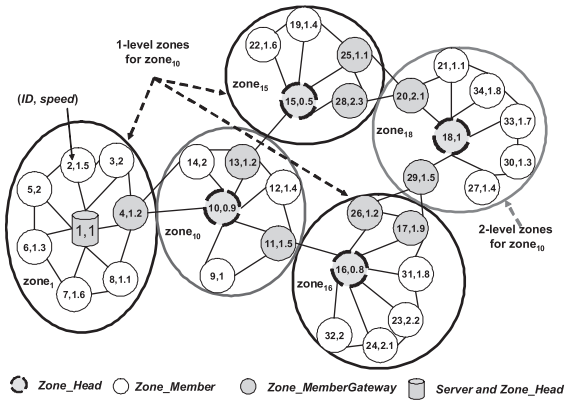



**Fig. 2** r-level neighboring zones.

In HDAR, the servers exchange some information with each other to know the current service demand in the entire network. This information will help to choose an appropriate number of new service replicas. In such dynamic network, it is not suitable to exchange information among all servers because of the wastage of network resources. To meet this challenge in a suitable manner, HDAR introduces $r$-level coverage confirmation mechanism where $r$ is a control parameter which determines the level of a neighboring server (will be described later in Sect. 4.3.1).

In HDAR, the replication mechanism depends on three factors: (i) the number of received service requests from all clients in each active zone $sq(s, zone_{zh}, t)$, (ii) the number of nodes in this active zone $n_{zone_{zh}}(t)$, and (iii) the tradeoff between communication and replication costs for this active zone. The first two factors with the received confirmation messages information determine the *degree of interest* of every active zone $zone_{zh}$ which is denoted by $DoI(s, zone_{zh}, t)$. Based on $DoI(s, zone_{zh}, t)$ and the tradeoff between communication and replication costs, server $s$ selects new replica zones. In the next subsections, first, we will explain the coverage confirmation mechanism and how HDAR determines for each zone the levels of other zones. Secondly, we will show how HDAR determines the value of $DoI(s, zone_{zh}, t)$. Finally, we will show how HDAR estimates the communication and replication costs for each active zone.

### 4.3.1 Zone Level and Coverage Confirmation Mechanism

In zone-based architecture, every zone head $zh$ has a set of nodes with *Zone_MemberGeteway* state. From hello messages received from this set of gateway nodes, each $zh$ can know all neighboring zones. In HDAR, this set of neighboring zones is called *1-level zones* of $zh$. For each 1-level zone of $zh$, the set of its neighboring zones, which are not neighbors of $zh$, is called *2-level zones* of $zh$. For each $h \geq 3$, *h-level zones* of $zh$ are defined similarly. For example, as shown in Fig. 2, the 1-level zones of $zone_{10}$ are $zone_1$, $zone_{15}$, and $zone_{16}$, while its 2-level zone is $zone_{18}$.

In HDAR, each server $s$ sends a confirmation message to all neighboring servers of at most $r$-level as follows: (a) the server $s$ sends the message to its zone head $zh$, (b) $zh$ uses its set of gateway nodes to send this message to 1-level neighboring zones, (c) each zone head $zh'$ in 1-level zones received the message, will decrease $r$ by 1 and sends the message to its 1-level zones, and (d) steps (b) and (c) will be repeated until $r$ is equal to 0. Note that, any zone head discards the same message if it is received again. The confirmation message contains: its id, its zone id, its received service demand, the number of its clients, and the number of its active zones. By using the received confirmation messages, server $s$ constructs a *Confirmation Table (CFT)* which stores the following information of each neighboring server: its id, its received service demand, the number of its clients, the number of its active zones, and its server level as shown in Fig. 3.

| Server id | Aggregated Service demand | # of clients | # of active zones | Server level |
|---|---|---|---|---|
| $sid$ | $agSd_{sid}(t)$ | $tClients_{sid}(t)$ | $aZones_{sid}(t)$ | $level_{sid}(t)$ |

**Fig. 3** Confirmation Table (CFT) fields.

#### 4.3.2 Service Demand Level of Active Zone

First, server $s$ determines the average number of nodes for all active zones at time $t$ which is denoted by $an_s(t)$ and computed by:

$$an_s(t) = \frac{\sum_{zone_{zh} \in ART} n_{zone_{zh}}(t) + \sum_{sid \in CFT} tClients_{sid}(t)}{nz_s(t) + nf_s(t)} \quad (6)$$

where $nz_s(t)$ is the total number of active zones (the number of records in $ART$) that access $s$ and $nf_s(t)$ is the total number of active zones which access the neighboring servers of at most $r$-level.

Secondly, we compute the service demand level, $sdl(s, zone_{zh}, t)$, for each active $zone_{zh}$ which accesses $s$ as follows:

$$sdl(s, zone_{zh}, t) = \frac{sq(s, zone_{zh}, t)}{an_s(t)} \quad (7)$$

Finally, $DoI(s, zone_{zh}, t)$ is defined as:

$$DoI(s, zone_{zh}, t) = \begin{cases} \text{high} & \text{if } sdl(s, zone_{zh}, t) \geq SDL_{th}, \\ \text{low} & \text{if } sdl(s, zone_{zh}, t) < SDL_{th} \end{cases} \quad (8)$$

where, $SDL_{th}$ is a predefined a service demand level threshold.

#### 4.3.3 Communication and Replication Costs Estimation

In HDAR, each server $s$ uses its $ART$ contents to estimate the communication and replication costs for each active zone $zh$. The server $s$ estimates the communication cost of the active zone $zone_{zh}$ by the following equation:

$$com(s, zone_{zh}, t) = (E_{tr} + E_{rx}) \times sq(s, zone_{zh}, t) \times [d(zone_{zh}, s, t) + d(s, zone_{zh}, t)] \quad (9)$$

and estimates the replication cost of the active zone $zh$ by the following equation:

$$rep(s, zone_{zh}, t) = k \times (E_{tr} + E_{rx}) \times d(s, zone_{zh}, t) \quad (10)$$

where $k$ is the number of service packets (program and data).

#### 4.3.4 HDAR Replication Rule

Based on $DoI(s, zone_{zh}, t)$ and the estimation of communication and replication costs, we propose HDAR replication mechanism. In this mechanism, every server $s$ selects multiple active zones without replicas to host new replicas once the replication process is triggered. These active zones with replicas are determined based on the following rule:

"*Replicate a service hosted at a server s to an active zone* $zone_{zh}$ *if* $DoI(s, zone_{zh}, t)$ *is high or if* $rep(s, zone_{zh}, t)$ *is less than* $com(s, zone_{zh}, t) \times RI$."

In this rule, the new replica zones are selected if: (i) their degree of interest is high which means that the service is highly demanded in these zones or (ii) the estimation of their replication costs are lower than their communication costs, which means that the new replicas will be resulted in lower cost compared to the communication cost.

Algorithm 1 shows the behavior of servers to realize the proposed replication mechanism. Every active server $s$ executes this algorithm every replication interval $RI$ based on its $ART$ and $CFT$ contents or the tradeoff between the communication and the replication costs to select new replica zones. In lines 1 to 4, the algorithm initializes the following variables: the number of active zones in $ART$, $an_s(t)$, the number of active zones in $CFT$, $nf_s(t)$, the total nodes in $ART$, $artNodes$, and the total nodes in $CFT$, $cftNodes$. In lines 5 to 8, the algorithm calculates the number of active zones $nz_s(t)$ and the total number of nodes in all active zones which exist in $ART$. In lines 9 to 12, the algorithm calculates the number of active zones $nf_s(t)$ and the total number of nodes in all active zones which exist in $CFT$. In line 13, the algorithm calculates the average number of nodes in each zone, $an_s(t)$. In lines 14 to 25, the algorithm determines active zones which will host new service replicas as follows: (i) it calculates the average number of received service requests, $sdl(s, zone_{zh}, t)$, by using the number of received service requests sent from each zone, $sq(s, zone_{zh}, t)$, for each active

**Algorithm 1** *HDAR*

**/*Replication mechanism at server $s$*/**

1: $nz_s(t) = 0$
2: $nf_s(t) = 0$
3: $artNodes = 0$
4: $cftNodes = 0$
5: **for each** $zone_{zh} \in ART$ **do**
6: $\quad artNodes = artNodes + n_{zone_{zh}}$
7: $\quad nz_s(t) = nz_s(t) + 1$
8: **end for**
9: **for each** $sid \in CFT$ **do**
10: $\quad cftNodes = cftNodes + tClients_{sid}(t)$
11: $\quad nf_s(t) = nf_s(t) + aZones_{sid}(t)$
12: **end for**
13: $an_s(t) = (artNodes + cftNodes)/(nz_s(t) + nf_s(t))$
14: **for each** $zone_{zh} \in ART$ **do**
15: $\quad sdl(s, zone_{zh}, t) = sq(s, zone_{zh}, t)/an_s(t)$
16: $\quad$ **if** $sdl(s, zone_{zh}, t) \geq SDL_{th}$ **then**
17: $\quad\quad DoI(s, zone_{zh}, t) = high$
18: $\quad$ **else**
19: $\quad\quad DoI(s, zone_{zh}, t) = low$
20: $\quad$ **end if**
21: $\quad$ **if** $DoI(s, zone_{zh}, t) = high$ **or** $rep(s, zone_{zh}, t) < com(s, zone_{zh}, t) \times RI$ **then**
22: $\quad\quad replicateService(s, zone_{zh})$
23: $\quad$ **end if**
24: **end for**

**Algorithm 2**

**/*Termination mechanism at zone head $zh$*/**

1: **if** $|SS_{zone_{zh}}| > 1$ **then**
2: $bs = bestServer(SS_{zone_{zh}})$
3: **for each** $s \in SS_{zone_{zh}}$ **do**
4: **if** $s \neq bs$ **then**
5: $shutdown(s)$
6: **end if**
7: **end for**
8: **end if**

zone $zone_{zh}$ and (ii) based on $sdl(s, zone_{zh}, t)$ values, the algorithm determines which active zones have a high degree of interest as shown in $if$ statement in line 16. As a result, if $sdl(s, zone_{zh}, t)$ is larger than or equal to $SDL_{th}$, then $DoI(s, zone_{zh}, t)$ of this active zone $zone_{zh}$ is high and the service is replicated to this active zone. If $DoI(s, zone_{zh}, t)$ of this active zone is low, the algorithm replicates the service to this active zone if its $com(s, zone_{zh}, t)$ is larger than its $rep(s, zone_{zh}, t)$ as shown in lines 21 to 23. Otherwise, the service is not replicated. Algorithm 2 shows the termination mechanism at zone head $zh$ which is described as follows: in line 1, the zone head $zh$ checks the number of servers in its zone. Here, $|SS_{zone_{zh}}|$ denotes the number of servers in zone $zone_{zh}$. If there are more than one server, the zone head $zh$ selects the best server based on server's resources (e.g. residual energy, available memory, number of neighbors, etc.) as shown in line 2. In lines 3 to 5, the zone head $zh$ terminates the other servers.

As an example to show the replication mechanism, assume that there is a service at node 1 (in 1's zone as shown in Fig. 1) and its $ART$ as shown in Table 2 at time $t$. We suppose that the number of service packets (program and data) $k$ is 4 and that $E_{tr}$ and $E_{rx}$ are 1.5 mW and 0.9 mW, respectively. Assume that $SDL_{th}$ is 1. As shown in Fig. 1 (a), $zone_{10}$, $zone_{16}$ and $zone_{18}$ are three active zones for the server 1 (i.e. $nz_1 = 3$) and $zone_{15}$ is a passive zone.

So, we have,

$$an_1(t) = \frac{n_{zone_{10}}(t) + n_{zone_{16}}(t) + n_{zone_{18}}(t)}{nz_1(t) + nf_1(t)} = \frac{6+7+8}{3+0} = 7 \quad (11)$$

where the algorithm calculates the average density, $an_1$, of all active zone for the server 1 (here $nf_1$ is equal to 0, because there is no any neighboring servers to server, 1). After that, by using Eq. (11), the algorithm calculates the average number of received service requests and estimates the communication and replication costs for each active zone by using Eqs (7), (9), and (10) as shown in Table 3.

By using the calculated values in Table 3, the algorithm determines the degree of interest for each active zone as follows:

$$DoI(1, zone_{10}, t) = low \quad (12)$$

$$DoI(1, zone_{16}, t) = low \quad (13)$$

$$DoI(1, zone_{18}, t) = high \quad (14)$$

**Table 3** Calculation parameters table.

| $s$ | $zone_{zh}$ | $sdl(s, zone_{zh}, t)$ | $com(s, zone_{zh}, t)$ | $rep(s, zone_{zh}, t)$ |
|---|---|---|---|---|
| 1 | $zone_{10}$ | $\frac{6}{7} < 1$ | 28.8 | 19.2 |
| 1 | $zone_{16}$ | $\frac{3}{7} < 1$ | 28.8 | 38.4 |
| 1 | $zone_{18}$ | $\frac{10}{7} > 1$ | 168 | 67.2 |

Finally, from Eqs. (12), (13), and (14), HDAR selects $zone_{10}$ and $zone_{18}$ to host new replicas because $DoI(1, zone_{18}, t)$ is high and $DoI(1, zone_{10}, t)$ is low but its value of $com(1, zone_{10}, t) \times RI$ is larger than its $rep(1, zone_{10}, t)$ as shown in Table 3. As a result, the network will contain three servers (1, 10 and 18) as shown in Fig. 1 (b). In the next step, each server will execute the replication algorithm based on its $ART$ and $CFT$ contents to select new replica zones. As shown in Fig. 1 (b), if we assume that the active zone of server 10 is $zone_{16}$ and active zone of server 18 is $zone_{15}$. This means that $ART$ of server 1 is empty, $ART$ of server 10 contains $zone_{16}$, and $ART$ of server 18 contains $zone_{15}$. If we assume that each server sends a confirmation message up to 1-level neighboring servers (i.e. $r$=1). This means that $CFT$ of server 1 contains information for server 10, $CFT$ of server 10 contains information for server 1, and $CFT$ of server 18 is empty. In this case, servers 1, 10, and 18 execute HDAR mechanism by using contents of their $ART$s and $CFT$s. As a result, server 1 does not replicate the service because there is no active zone with high degree of interest, server 10 may make a decision of replicating a new service into $zone_{16}$, and server 18 may make a decision of replicating a new service into $zone_{15}$, in the future.

## 5. Performance Evaluation

In order to evaluate the effectiveness of HDAR, we compared its performance with SDP [10] and DAR [11] for the following metrics:

**Energy consumption**: energy consumption is the summation of (a) Service cost: energy amounts consumed for communication and replication defined by objective function (4) and (b) Confirmation cost: energy amounts consumed for confirmation messages and are defined as follows:

$$TotalFCost(S(t), t) = \sum_{s \in S(t)} FrmCost(s, NigS_r(s, t), t) \quad (15)$$

where

$$FrmCost(s, NigS_r(s, t), t) = (E_{tr} + E_{rx}) \times \sum_{s' \in NigS_r(s,t)} d(s, s', t) \quad (16)$$

and $NigS_r(s, t)$ is the set of neighboring servers of $s$ at level less than or equal to $r$. Note that, there is no confirmation cost in case of DAR and SDP.

**Table 4** Configuration parameters.

| Configuration parameter | Value in simulation |
|---|---|
| Number of nodes | 25 to 200 |
| Maximum node's speed | [1...10] meters/second |
| Field size | $500\,m \times 500\,m$ |
| Transmission range | 100 m |
| Bandwidth | 2 *Mbps* wireless channel |
| Routing Protocol | *Ad-hoc On-Demand Distance Vector Routing* (AODV) [18] |
| MAC Protocol | IEEE 802.11b without power control protocol |
| $E_{tr}$ | $1.5\,mW$ |
| $E_{rx}$ | $0.9\,mW$ |
| Simulation time | 700 seconds |
| RI | {40, 50, 60, 70 ,80} seconds |

**Replica ratio**: ratio of the number of service providers at each point of time during the network lifetime to the total number of nodes in the network. In order to avoid wasting the network resources, a low replica ratio together with high service availability and low energy consumption are desired.
**Service availability**: we use the same definition of service availability as in [7], [10], that is, the ratio of the number of service replies received to the number of service requests sent during the network lifetime.

The QUALNET [19] simulator was used with input parameters as listed in Table 4, such as network size, speed, transmission range, simulation time, etc. In addition, the node mobility was based on two mobility models: (i) Random way point (RWP) mobility model [20] where the minimum speed was 0 *meter/second*, the pause time was 0 *second* (continuous movement) and the maximum speed between [1...10] *meters/second* and (ii) Realistic mobility model generated by MobiREAL [21] which is a simulator to model and simulate realistic mobility of nodes. To simulate client's requests, each client maintained a requesting rate according to Poisson distribution with average $\lambda$ which equals to 6 requests in time interval of 60 seconds.

Initially, there was one service with 2 *Kbytes* of object size (program and data) and the size of one packet is 512 *bytes* (so, the number of service packets $k$ was 4) which was put at a certain node. For HDAR, we experimented three cases by using several different values for $r$ and are denoted by *HDAR-{1,2,5}*. For SDP, we experimented two cases by using several different replication and hibernation thresholds which are denoted by *SDP-RH 4/5-1*, we used both hibernation and replication mechanisms with 1 and 4 or 5 requests as hibernation and replication thresholds, respectively. We repeated every simulation 5 times then averaged the results.

From preliminary experiments, we decided that $SDL_{th} = 1$ for a good trade-off between service availability and energy consumption.

### 5.1 Random Way Point Scenario

In order to show the performance of HDAR, we compared HDAR with SDP [10] and DAR in terms of replica ratio, service availability, and energy consumption against replication interval, maximum node's speed, and network size, as well as the average hop counts against network size. We show the results in Figs. 4 to 13.

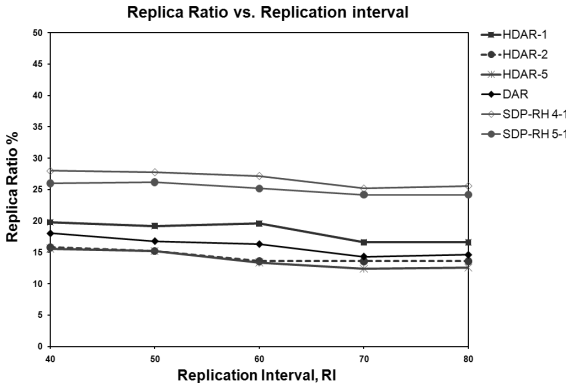



**Fig. 4** Replica ratio vs. replication interval.

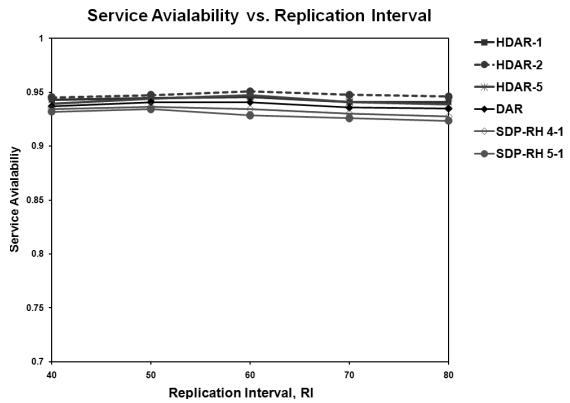



**Fig. 5** Service availability vs. replication interval.

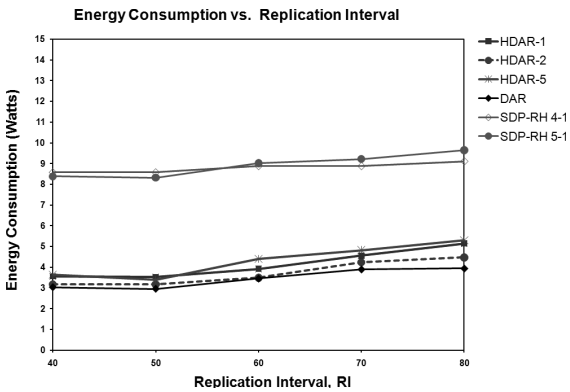



**Fig. 6** Energy consumption vs. replication interval.

#### 5.1.1 Replication Interval Effects

Figures 4, 5, and 6 show the replica ratio, the service availability, and the energy consumption, respectively, against the replication interval when maximum node's speed was 1 *meter/second*, and network size was 100.

**1) Replica ratio and service availability:** As shown in Fig. 4, the replica ratio of all algorithms decreases as the replication interval increases. This is because, the number of executed replication processes to create new replicas decreases. The replica ratio for SDP was higher than DAR and HDAR. In addition, the replica ratio for SDP was affected by values of replication and hibernation thresholds. That is, the number of replicas decreased as the replication threshold increased. In case of HDAR, the replica ratio is less affected by the replication interval when $r$ was 2. As shown in Fig. 5, the service availability of all algorithms were less affected when the replication interval increased. The service availability of HDAR was higher than DAR and SDP and was between 0.94 and 0.95. In case of DAR, the service availability was between 0.93 and 0.94, while the service availability of SDP was between 0.92 and 0.93 when replication threshold was 4 requests. Also, the service availability of SDP was affected by replication and hibernation thresholds and decreased as the replication threshold increased.

**2) Energy consumption:** As shown in Fig. 6, the energy consumption increased as expected as the replication interval increased. This is because, when the replication interval increases, the communication cost increases. The energy consumption for DAR was much lower than SDP. In addition, the energy consumption for SDP was affected by values of replication and hibernation thresholds. In case of HDAR, the energy consumption was much lower than SDP but higher than DAR. This is because, HDAR consumes additional energy for confirmation messages between servers for different values of $r$. When $r$ was 2, the energy consumption was less than other values of $r$.

As a result, when $r$ was 2, HDAR achieved higher service availability than DAR and SDP with reasonable energy consumption and less affected by the replication interval. In addition, the replica ratio was less affected than other values of $r$.

### 5.1.2 Maximum Node's Speed Effects

Figures 7, 8, and 9 show the replica ratio, the service availability, and the energy consumption, respectively, against the maximum node's speed when the network size was 100 and the replication interval was 70 seconds.

**1) Replica ratio and service availability:** As shown in Fig. 7, the replica ratio of all algorithms decreased as the maximum node's speed increased. For low speed (from 1 to 5 *meters/second*), the replica ratio decreased slowly. For high speed (from 6 to 10 *meters/second*), the replica ratio decreased quickly. As shown in Fig. 8, the service availability of all algorithms decreased as the maximum node's speed increased. For low speed (from 1 to 5 *meters/second*), the service availability decreased slowly. For high speed (from 6 to 10 *meters/second*), the service availability decreased quickly. However, HDAR achieved higher service availability than DAR and SDP. This is because, HDAR considers the service demand at neighboring servers and the tradeoff between the communication and replication costs, while DAR and SDP do not. In addition, when $r$ was 2, HDAR achieved higher service availability than other values of $r$.

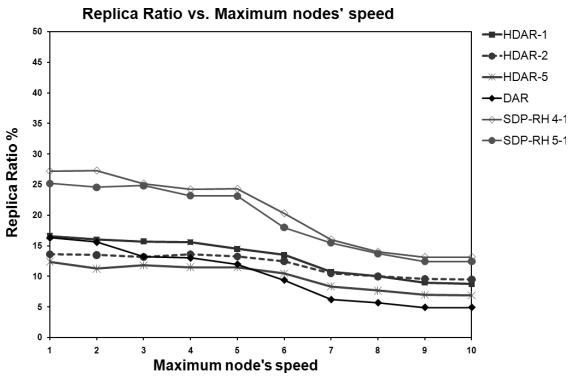



**Fig. 7** Replica ratio vs. maximum node's speed.

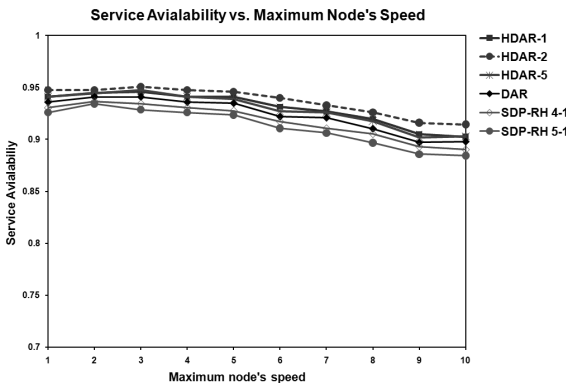



**Fig. 8** Service availability vs. maximum node's speed.

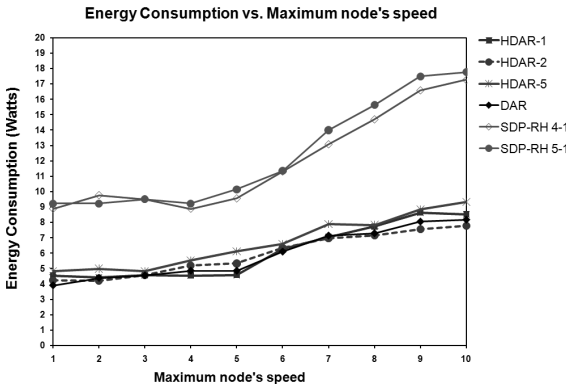



**Fig. 9** Energy consumption vs. maximum node's speed.

**2) Energy consumption:** As shown in Fig. 9, the energy consumption increased as the maximum node's speed increased. This is because, when the maximum node's speed increases, the locations of replicas changes rapidly and the number of replicas decreases. So, the communication cost increases. The energy consumption for DAR and HDAR were much lower than SDP. In addition, the energy con-

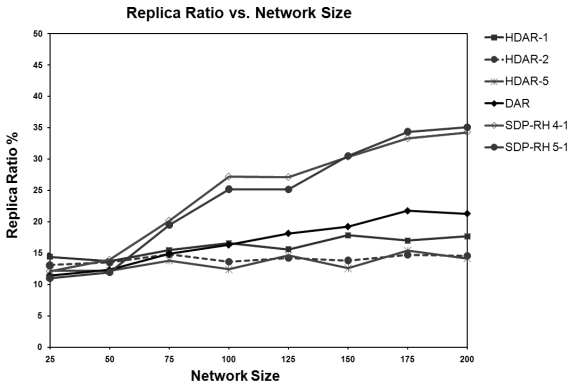



**Fig. 10** Replica ratio vs. network size.

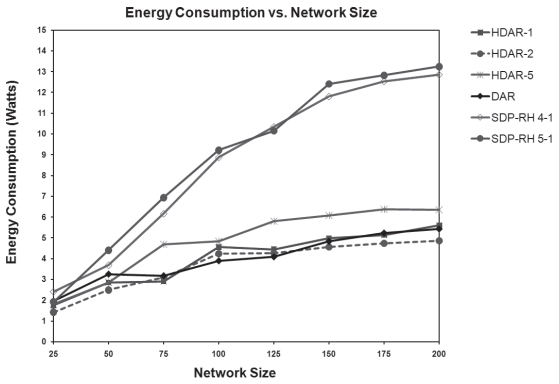



**Fig. 12** Energy consumption vs. network size.

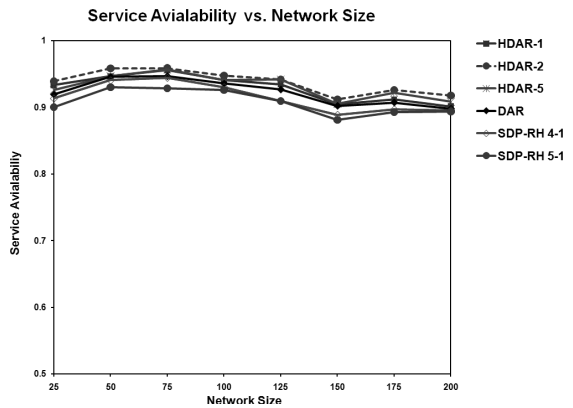



**Fig. 11** Service availability vs. network size.

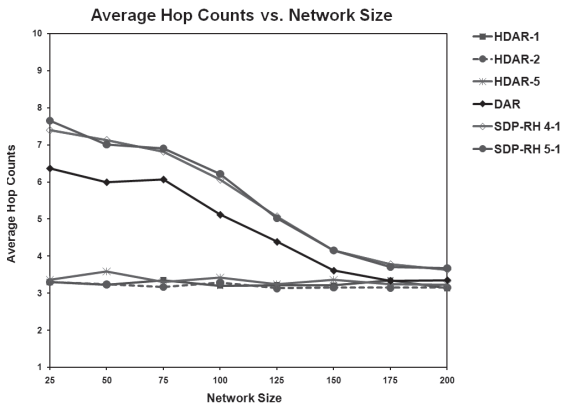



**Fig. 13** Average hop counts vs. network size.

sumption for SDP was affected by values of replication and hibernation thresholds. In case of HDAR, the energy consumption was much lower than SDP but higher than DAR. This is because, HDAR consumes additional energy for confirmation messages between servers for different values of $r$.

As a result, when $r$ was 2, HDAR achieved higher service availability than DAR and SDP with reasonable energy consumption.

### 5.1.3 Network Size Effects

Figures 10, 11, 12, and 13 show the replica ratio, the service availability, the energy consumption, and the average hop counts, respectively, against the network size when the maximum nod's speed was 1 *meter/second* and the replication interval was 70 seconds.

**1) Replica ratio and service availability:** As shown in Fig. 10, the replica ratio of DAR and SDP increased as the network size increased. This is because, when the number of nodes increased, the number of requests increased and new replicas were created. However, DAR showed better scalability than SDP. In addition, the replica ratio for SDP was affected by values of replication and hibernation thresholds. That is, the number of replicas decreased as the replication threshold increased. On the other hand, the replica ratio of HDAR was fluctuated but almost independent of the network size. This is because, HDAR uses the confirmation messages between servers and considers the tradeoff between the communication and the replication costs for each zone. Also, when $r$ was 2 the replica ratio is less fluctuated than other values of $r$. As shown in Fig. 11, the service availability of DAR was higher than SDP and the service availability of SDP was affected by replication and hibernation thresholds and decreased as the replication threshold increased. On the other hand, the service availability of HDAR was higher than DAR and SDP, because HDAR considers the service demand at the neighboring servers and the tradeoff between the communication and replication costs, while DAR and SDP do not.

**2) Energy consumption:** As shown in Fig. 12, the energy consumption increased as expected as the network size increased. This is because when the number of nodes increased, the number of requests increased and new replicas were created with additional cost. The energy consumption for DAR was scalable and kept the energy consumption much lower than SDP. In addition, the energy consumption for SDP was affected by values of replication and hibernation thresholds. In case of HDAR, the energy consumption was much lower than SDP but higher than DAR. This is because, HDAR consumes additional energy for confirmation messages between servers for different values of $r$. When $r$

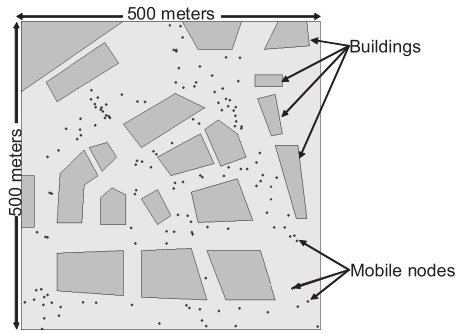



**Fig. 14** Urban map for MobiReal simulator.

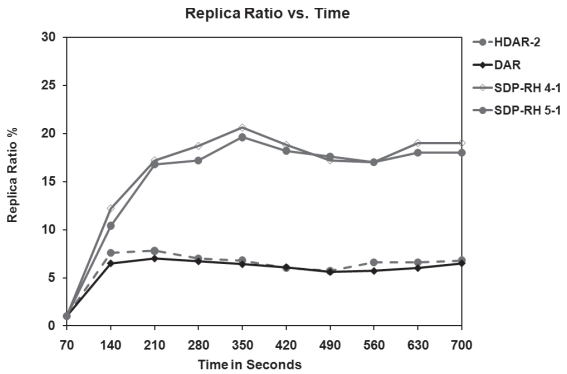



**Fig. 15** Replica ratio vs. time.

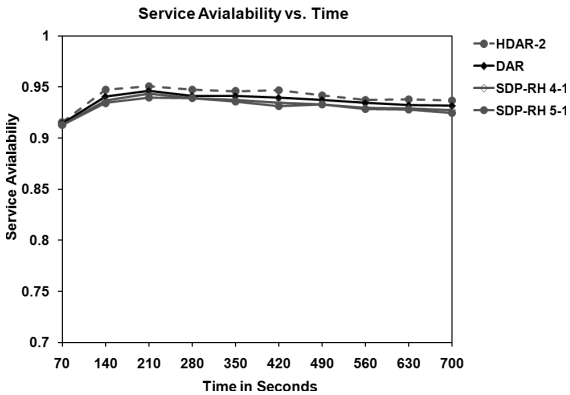



**Fig. 16** Service availability vs. time.

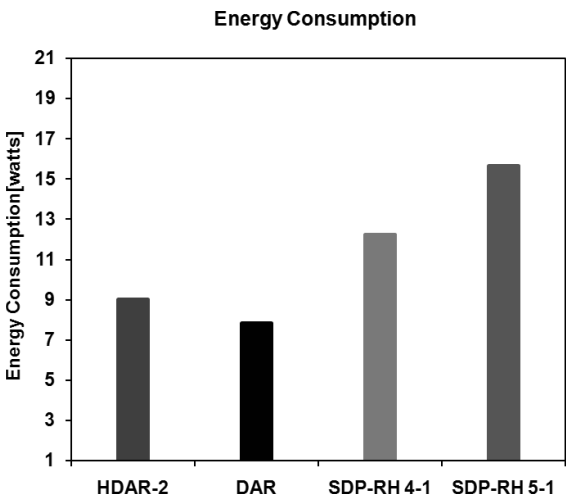



**Fig. 17** Energy consumption through network lifetime.

was 2, the energy consumption was almost less than other values of *r*.

As a result, when *r* was 2, HDAR achieved higher service availability than DAR and SDP with reasonable energy consumption. In addition, the replica ratio was less fluctuated than other values of *r*.

**3) Average hop counts:** As shown in Fig. 13, the average hop counts of DAR and SDP decreased as the network size increased. This is because, when the number of nodes increased, the number of replicas increased and the average hop counts decreases. Also, the average hop counts of DAR was lower than SDP. On the other hand, the average hop counts for HDAR was lower than DAR and SDP and was independent of the network size. This is because, HDAR considers the service demand at the neighboring servers and the tradeoff between the communication and replication costs, while DAR and SDP do not.

### 5.2 Realistic Mobility Scenario

In this scenario, we used MobiREAL simulator [21] to generate a realistic mobility model where 100 nodes were distributed in the simulation field as shown in Fig. 14. MobiREAL is a simulator to reproduce the user traffic based on the observation of the actual traffic density at each street. This simulator allows to describe how mobile nodes change their destinations, routes and speeds/directions based on their positions, surroundings such as obstacles (e.g. buildings) and neighboring nodes. In our simulations, realistic mobility means that the movement of mobile nodes in realistic environment through the incorporation of obstacles and the construction of realistic movement paths.

In order to show the effect of the realistic mobility[†], we compared the proposed protocol with DAR and SDP when network size was 100 nodes, *r* was 2 (the best value for *r* in case of random way point mobility model), the replication interval was 70 seconds, the initial speed was 1 *meter/second*, and the other parameters were the same as in random way point model. For SDP, we experimented for two cases, SDP-RH 4/5-1, that were defined in the previous section. We show the results in Fig. 15, Fig. 16, and Fig. 17.

#### 5.2.1 Replica Ratio and Service Availability

As shown in Fig. 15, the replica ratio of SDP fluctuated overtime. While, for HDAR and DAR the replica ratio was more stable than SDP. In addition, the replica ratio for SDP was

[†]We did not consider the effect of obstacles such as buildings on the radio strength between nodes.

affected by values of replication and hibernation thresholds. As shown in Fig. 16, the service availability of HDAR was higher than DAR and SDP and was between 0.91 and 0.95.

#### 5.2.2 Energy Consumption

As shown in Fig. 17, the energy consumption for DAR was much lower than SDP. In addition, the energy consumption for SDP was affected by values of replication and hibernation thresholds. In case of HDAR, the energy consumption was much lower than SDP but higher than DAR. This is because, HDAR consumes additional energy for confirmation messages between servers where DAR did not use such messages.

As a result, HDAR achieved higher service availability than DAR and SDP with reasonable energy consumption.

## 6. Conclusion

In this paper, a new distributed adaptive service replication method for MANETs was presented. Our protocol first divides the whole network into disjoint zones with diameters of at most 2 hops, selects a node with minimum moving speed in each zone as a zone head, and constructs a virtual backbone network connecting all zone heads. By using this zone structure, our protocol selects a new replica node according to the topology, number of requests from clients, and the tradeoff between communication an replication costs for each zone. Our protocol is scalable and it can control the locations and the number of service replicas, keeping the network-wide energy consumption as low as possible and improving the service availability. Simulations demonstrated that our method improves the performance of service provision in terms of the energy consumption and service availability compared with existing methods. In addition, the path length between a client and a server is minimized independent of network size.

In the future work, we will improve our distributed service replication algorithm to maintain synchronization and consistency of service replicas. Also, we will consider the presence of selfish nodes and its effect on the network performance. In addition, we will implement the proposed method and experiment in the real world to know its performance under realistic conditions.

### References

[1] Web Services and Service-Oriented Architectures, http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html

[2] S. Dustdar and L. Juszczyk, "Dynamic replication and synchronization of web services for high availability in mobile ad-hoc networks," Service Oriented Computing and Applications, vol.1, no.1, pp.19–33, April 2007.

[3] L. Juszczyk, "Replication and synchronization of web services in ad-hoc networks," Masters thesis, Technischen University at Wien, 2005.

[4] M. Hauspie, D. Simplot, and J. Carle, "Partition detection in mobile ad hoc networks using multiple disjoint paths set," Proc. 2nd Mediterranean Workshop on Ad-Hoc Networks, June 2003.

[5] K. Wang and B. Li, "Efficient and guaranteed service coverage in partitionable mobile ad-hoc networks," Proc. IEEE INFOCOM'02, vol.1, pp.1089–1098, June 2002.

[6] B. Li and K.H. Wang, "Nonstop: Continuous multimedia streaming in wireless ad hoc networks with node mobility," IEEE J. Sel. Areas Commun., vol.21, no.10, pp.1627–1641, Dec. 2003.

[7] A. Derhab, N. Badache, and A. Bouabdallah, "A partition prediction algorithm for service replication in mobile ad hoc networks," Proc. 2nd Conf. on Wireless On-demand Network Systems and Services, pp.236–245, Jan. 2005.

[8] A. Derhab and N. Badache, "A pull-based service replication protocol in mobile ad hoc networks," European Trans. Telecommunications, vol.18, no.1, pp.1–11, Oct. 2005.

[9] C.A. Bellavista and P.E. Magistretti, "REDMAN: An optimistic replication middleware for read-only resources in dense MANETs," Elsevier J. Pervasive and Mobile Computing, vol.1, no.3, pp.279–310, Aug. 2005.

[10] M. Hamdy and B. Konig-Ries, "A service distribution protocol for mobile ad hoc networks," Proc. 5th Int'l. Conf. on Pervasive Services, pp.141–146, July 2008.

[11] A. Ahmed, K. Yasumoto, N. Shibata, T. Kitani, and M. Ito, "DAR: Distributed adaptive service replication for MANETs," Proc. 5th IEEE Int'l. Conf. on Wirless Mobile Computing, Networking, and Communications, pp.91–97, Oct. 2009.

[12] V.D. Park and M.S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," Proc. IEEE INFOCOM'97, pp.1405–1413, April 1997.

[13] S.E. Athanaileas, C.N. Ververidis, and G.C. Polyzos, "Optimized service selection for MANETs using an AODV-based service discovery protocol," Proc. 6th Mediterranean Ad Hoc Networking Workshop, pp.9–16, June 2007.

[14] G. Cornuejols, G. Nemhauser, and L. Wolsey, "The uncapacitated facility location problem," in Discrete Location Theory, ed. P. Mirchandani and R. Francis, pp.119–171, John Wiley & Sons, 1990.

[15] R. Ramanathan and M. Steenstrup, "Hierarchically-orgainsed multihop mobile networks for qulaity-of-service support," Mobile Netw. Appl., vol.3, no.2, pp.101–119, June 1998.

[16] A. Ephremides, J.E. Wieselthier, and D.J. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling," Proc. IEEE, vol.75, no.1, pp.56–73, Jan. 1987.

[17] M. Gerla and J.T. Tsai, "Multicluster, mobile, multimedia radio network," ACM J. Wireless Networks, vol.1, no.3, pp.255–265, Oct. 1995.

[18] C.E. Perkins, E.M. Royer, and S.R. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF Internet Draft, Manet Working Group, draft-ietf-manet-aodv-10.txt, Jan. 2002.

[19] Qualnet 4.0 network simulator, Scalable Network Technologies, 2007.

[20] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," Mobile Computing, pp.153–181, 1996.

[21] K. Maeda, A. Uchiyama, T. Umedu, H. Yamaguchi, K. Yasumoto, and T. Higashino, "Urban pedestrian mobility for mobile wireless network simulation," Int'l. Elsevier Journal on Ad hoc Networks, vol.7, no.1, pp.153–170, Jan. 2009.

**Asaad Ahmed** received his B.S. degree in Computer Science from Faculty of Science, Alexandria University, Egypt in 1999. He received M.S. in Computer Science from Faculty of Science, Cairo University, Egypt in 2008. Currently, he is a Ph.D. student candidate at Graduate School of Information Science, Nara Institute of Science and Technology, Japan. His current research interests include distributed systems, mobile ad hoc networks and ubiquitous computing.

**Keiichi Yasumoto** received the B.E., M.E., and Ph.D. degrees in Information and Computer Sciences from Osaka University, Osaka, Japan, in 1991, 1993 and 1996, respectively. He joined the faculty of Shiga University in 1995. Since 2002 he has been an Associate Professor of Graduate School of Information Science at Nara Institute of Science and Technology. His current research interests include mobile computing, ubiquitous computing, and multimedia communication. He is a member of IPSJ, ACM and IEEE/CS.

**Minoru Ito** received the B.E., M.E., and Ph.D. degrees in Information and Computer Sciences from Osaka University in 1977, 1979 and 1983, respectively. He joined the faculty of Osaka University in 1979. Since 1993, he has been a professor of Graduate School of Information Science at Nara Institute of Science and Technology. He is a member of IPSJ, ACM and IEEE.

**Naoki Shibata** received the M.E. and Ph.D. degrees in Information and Computer Sciences from Osaka University, Osaka, Japan, in 1998, 2001, respectively. He joined the faculty of Nara Institute of Science and Technology in 2001. Since 2004, he has been an Associate Professor of Shiga University. His current research interests include distributed systems, ubiquitous computing, and multimedia communication. He is a member of IPSJ, ACM and IEEE/CS.

**Tomoya Kitani** received his B.E., M.E., and Ph.D. degrees in Information Science and Technology from Osaka University in 2002, 2004 and 2006, respectively. From 2005 to 2008, he was an assistant professor at Graduate School of Information Science, Nara Institute of Science and Technology. Since 2008 he has been serving as an assistant professor of at the Division of Global Research Leaders, Shizuoka University. His research interests include combinatorial problems, embedded systems and vehicular ad hoc networks. He is a member of IEEE and IPSJ.