

# A survey of data replication techniques for mobile ad hoc network databases

Prasanna Padmanabhan · Le Gruenwald ·  
Anita Vallur · Mohammed Atiquzzaman

Received: 22 April 2006 / Revised: 14 November 2006 / Accepted: 29 March 2007 / Published online: 30 May 2008  
© Springer-Verlag 2008

**Abstract** A mobile ad hoc network (MANET) is a network that allows mobile servers and clients to communicate in the absence of a fixed infrastructure. MANET is a fast growing area of research as it finds use in a variety of applications. In order to facilitate efficient data access and update, databases are deployed on MANETs. These databases that operate on MANETs are referred to as MANET databases. Since data availability in MANETs is affected by the mobility and power constraints of the servers and clients, data in MANETs are replicated. A number of data replication techniques have been proposed for MANET databases. This paper identifies issues involved in MANET data replication and attempts to classify existing MANET data replication techniques based on the issues they address. The attributes of the replication techniques are also tabulated to facilitate a feature comparison of the existing MANET data replication works. Parameters and performance metrics are also presented to measure the performance of MANET replication techniques. In addition, this paper also proposes criteria for selecting appropriate data replication techniques for various application requirements. Finally, the paper concludes with a discussion on future research directions.

**Keywords** Mobile databases · Mobile ad hoc network databases · Data replication

---

P. Padmanabhan · L. Gruenwald (✉) · A. Vallur · M. Atiquzzaman  
The University of Oklahoma, School of Computer Science,  
Norman, OK 73019, USA  
e-mail: ggruenwald@ou.edu

P. Padmanabhan  
e-mail: prasannap@yahoo-inc.com

A. Vallur  
e-mail: vkanita@ou.edu

M. Atiquzzaman  
e-mail: atiq@ou.edu

## 1 Introduction

A mobile ad hoc network (MANET) is a collection of autonomous wireless nodes that may move unpredictably, forming a temporary network without any fixed backbone infrastructure [10, 19]. In such a network, each node not only plays the role of an end system, but also acts as a router that forwards packets to desired destination nodes. These nodes are capable of both single and multi-hop communication. Mobility and the absence of any fixed infrastructure make MANETs very attractive for military and rescue operations, sensor networks and time-critical applications [17].

In a distributed database system, data are often replicated to improve reliability and availability, thus increasing its dependability. In addition, data are also stored on computers where it is most frequently needed in order to reduce the cost of expensive remote access. However, a reliable system might not always ensure high availability [16]. An important issue facing data replication is the correctness of the replicated data (often termed as data *consistency*) that exist in different systems.

Communication failure between two or more mobile nodes in a MANET may lead to network partitioning, where the network is divided into isolated sub networks, giving rise to the possibility of data inconsistency. In addition to node mobility, slow response from certain nodes may also cause a network to “appear” partitioned, even when it is not actually partitioned. A survey of various techniques to maintain data consistency in partitioned fixed networks is given in [6]. Ensuring one-copy serializability after reconnection of partitions is a serious issue in distributed database systems. A decision has to be made to abort/commit transactions that are running during the occurrence of partitioning. Various issues and solutions associated with data replication for distributed databases in fixed networks are discussed in [8]. However,

to the best of our knowledge, there exists no survey of data replication techniques for MANET databases. The aim of this paper is to fill in this gap by providing a comprehensive review of existing data replication techniques for MANET databases. The results presented in this article will help database designers to select appropriate data replication techniques for MANET databases.

The remainder of this paper is organized as follows. Section 2 discusses the various issues concerning data replication in MANET databases. Based on the issues, Sect. 3 identifies the criteria that can be used to classify the existing replication techniques for MANET databases. Sections 4–8 review the current replication techniques using the established criteria. Section 9 presents a feature comparison of all the data replication techniques discussed in this paper. Section 10 provides guidelines for selecting a data replication technique for a MANET database application, and, parameters and metrics to measure the performance of a replication technique. Finally, Sect. 11 concludes the paper and identifies future research directions.

## 2 Issues concerning data replication for manet databases

In addition to addressing issues such as data consistency and availability that exist in traditional databases, a data replication technique for MANET databases must also deal with the following additional issues arising from constraints imposed by their specific environments and applications:

- *Server power consumption:* Servers in MANET run on battery power. Power consumption of servers that provide database management system (DBMS) services to potentially many clients should be minimized. Servers with higher power availability are expected to perform more work than those that have lower power. If a server has low power remaining and if it is replicated with many frequently accessed data items (hot data), then frequent data access requests for these hot data might drain its power soon. Servers with no power remaining would not be able to provide any more services. The replication algorithm should thus replicate data items in such a way that the power consumption of servers is reduced, and is balanced among all servers in the system.
- *Server mobility:* Servers in MANET are mobile and the speed at which the network topology changes is higher than that in conventional mobile databases. Due to their mobility, servers might sometimes move to a place where they cannot be reached by other servers or clients. The replication technique should avoid replicating hot data items in such isolated servers.
- *Client mobility:* Clients that query the servers can be mobile. Clients sometimes send their transactions to the

nearest servers to get a quicker response. The decision to replicate a data item in a particular server may be based on the access frequency of that data item on that server. Clients, after issuing their requests for data access to a server, might move to new positions after a certain interval of time, and they might send their query and update requests to the nearest servers from their new locations. Hence, the access frequencies must be dynamic in nature and the decision to replicate data items in appropriate servers must also be dynamic.

- *Client power:* Client machines also run using their battery power. Some clients like PDAs are more power restricted than servers. They are limited by the amount of energy they can use before their batteries need to be recharged. A client might lose its power rapidly if it waits for its transactions' results for a long time. The replication technique should be able to replicate data items in appropriate servers in such a way that client power consumption is reduced.
- *Real-time applications:* MANET applications like rescue and military operations are time-critical and may contain both firm and soft real-time transactions. Therefore, the replication technique should be able to deliver correct information before the expiry of transaction deadlines, taking into consideration both real-time firm and soft transaction types in order to reduce the number of transactions missing their deadlines.
- *Frequent disconnection of mobile hosts:* Mobile hosts often get disconnected from the network due to various factors like power failure or their mobility. In addition, some mobile users switch their units on and off regularly to save power, causing more network disconnections. Servers which hold the data cannot provide services if they are disconnected from other mobile hosts. Thus, ideally, the replication algorithm should be able to determine when a particular mobile host would be disconnected and, accordingly, replicate its data items in a different server to improve data accessibility.
- *Network partitioning:* Due to frequent disconnection of mobile hosts, network partitioning occurs more often in MANET databases than in traditional databases. Network partitioning is a severe problem in MANET when the server that contains the required data is isolated in a separate partition, thus reducing data accessibility to a large extent. Therefore, the replication technique should be able to determine the time at which network partitioning might occur and replicate data items beforehand.

## 3 Classification of existing data replication techniques

Following from the discussion of the issues that concern replication in MANET databases, we can identify *the issues*

of power consumption (by both clients and servers), real-time requirements of applications and network partitioning (due to client and server mobility as well as frequent disconnection) as three of the most important issues to be considered in the design of a data replication technique for MANETs. While these issues are also related to conventional networks, they become more severe in MANETs. Since MANETs allows for mobility of both servers and clients, network disconnection and partitioning occur frequently in MANETs. Power failure also occurs commonly in MANETs due to the limited battery capacity of mobile nodes. These occurrences prolong transaction execution time due to unavailability of remote resources. Furthermore, MANETs are commonly deployed for time-critical applications; thus, enabling the execution of transactions before the expiry of their deadlines is difficult in the face of such network conditions. Hence, these issues (namely, power consumption, real-time requirements of applications and network partitioning) should be considered when designing a replication technique for MANET databases. Based on these MANET data replication issues, we can define the following. *power-aware* techniques take into account the power consumption of clients and servers. Replication techniques that support transactions having deadlines and try to minimize the number of transactions that miss their deadlines are called *real-time-aware* techniques. Replication techniques that predict the occurrence of network partitioning and replicate data items accordingly ahead of time are called *partition-aware* techniques. An ideal replication strategy for MANETs would be power-aware, real-time-aware as well as partition-aware.

The existing data replication techniques can be classified into five categories based on different combinations of the above mentioned criteria. Non-power-aware, non-real-time-aware and non-partition-aware techniques are reviewed in Sect. 4. Power-aware, non-real-time-aware and Non-Partition-Aware techniques are discussed in Sect. 5, while power-aware, real-time-aware and non-partition-aware techniques are in Sect. 6. Non-power-aware, non-real-time-aware and partition-aware techniques are reviewed in Sect. 7, followed by power-aware, non-real-time-aware and partition-aware techniques in Sect. 8. These categories are useful to classify existing replication techniques based on the MANET replication issues they try to address. This classification is not meant to enumerate or describe all possible types of MANET replication techniques.

#### 4 Non-power-aware, non-real-time-aware and non-partition-aware techniques

Techniques that concentrate on improving data accessibility in MANET databases, but do not address the issues related to

client and server power, real-time transactions and network partitioning, are called non-power-aware, non-real-time-aware and non-partition-aware techniques. These techniques are described in this section.

##### 4.1 Information dissemination in partitionable MANETS [20]

Karumanchi et al. [20] assumes an environment of frequent update transactions and replication on all the servers. It considers an architecture that consists of servers which act as peers (and communicate with each other) and clients that query the servers for data that they require. Every client updates its current location and state to all other servers. This is done to enable every server to retrieve the latest position information of every server. The update information is propagated in such a way that nodes querying this update information get the most recent data, thereby reducing inconsistency and dirty read transactions. Each node first determines the time to send the update information. After determining when to update, each node then determines where to send these updates and which nodes to query for the information in such a way that it mitigates the impact of network partitioning. It uses a variation of a quorum-based scheme. Given a set of servers  $S$ , a quorum is a set of  $m$  intersecting subsets which completely compose  $S$ . This technique is targeted for applications where inaccuracy is preferred to no information at all. Hence, it allows dirty read transactions to be executed. It attempts to address the two issues involved in information dissemination in MANETS: when to send updates and where to send update and query transactions.

###### 4.1.1 When to send updates?

Four different strategies that determine the time to send updates were proposed:

- *Time-based strategy*: Updates are propagated at time intervals exponentially distributed with mean  $t$  units.
- *Time and location-based strategy*: This strategy is an extension of the time-based strategy. In addition to propagating updates at exponentially distributed time intervals with mean  $t$  units, the node propagating the update also stores its location when it last sent an update. After the update request succeeds, the node does not send any more update requests unless its location changes. This is done to avoid duplicate update messages from the same location.
- *Absolute connectivity-based strategy*: Updates are propagated from a node when a specific number of nodes get connected or disconnected from it since its last update.

- *Percentage connectivity-based strategy*: Updates are propagated from a node when a specific percentage of links connected to it changes since its last update.

The last two strategies best suit MANET applications as the update propagations take place based on the current topology of the network.

#### 4.1.2 Where to send update and query transactions?

Every node  $x$  maintains a list of nodes that it cannot reach. These nodes are stored in a data structure called  $DQL(x)$ , the disqualified list for  $x$ . Nodes in  $DQL(x)$  are periodically removed after a certain period of time, called the disqualification period. Node  $x$  first selects a quorum (assume  $S_1$ ) and then sends its update request to all the servers in that quorum. All servers that do not reply to  $x$  (denoted as  $S_1^1$ ) are added to  $DQL(x)$ . Now another node  $y$  might query the same data item from a different quorum (assume  $S_2$ ). Some servers (denoted as  $S_2^1$ ) might not be able to reply as they might be in a different partition. There will be at least one common server between the two quorums  $S_1$  and  $S_2$ , but there might not be a common server between  $(S_1 - S_1^1)$  and  $(S_2 - S_2^1)$ . When there is no common server between the updated quorum and the quorum in which the data are read, the query might end up with a dirty read operation. Hence, the three following techniques were proposed to reduce dirty reads.

##### 4.1.2.1 Select then eliminate (STE)

1. Node  $x$  selects a quorum randomly (assume  $S_i$ ).
2.  $x$  then sends the update/query request to all the reachable servers in  $S_i$ .
3. An update request is considered to be successful if at least one response is received from these servers within the timeout period  $T_{\text{timeout}}$ . If more than one non-null response is received for a read operation, the one with a higher time stamp is selected.
4. If no response is received from a server then that server is included in  $DQL(x)$  until the disqualification period is reached.

This process is repeated until there is a success or until all quorums have been selected.

##### 4.1.2.2 Eliminate then select (ETS)

1. Node  $x$  first eliminates all the quorums that have at least one server in  $DQL(x)$ .
2.  $x$  then randomly selects one of the remaining quorums and sends its update/query request to all servers in that quorum.

3. If it receives at least one acknowledgment in the case of an update and a non-null reply in the case of a query, the request is considered to be successful.
4. Servers that do not respond within the timeout period  $T_{\text{timeout}}$  are included in  $DQL(x)$ .

This process is repeated until there is a success or until all quorums have been selected.

*Hybrid strategy* The hybrid strategy makes use of both ETS and STE to take the advantage of these strategies to increase the number of servers receiving the updates, thereby resulting in accurate information retrieval by queries. It uses ETS for updates and STE for read queries in order to ensure that updates are propagated to a maximum possible number of nodes and that read queries always read recent data.

These techniques target applications where dirty reads are preferred to no information. It does not provide a scheme to allocate replicas; it only provides a means to access data sources and update them. Even with frequent node disconnections and changing network topology, the number of successful transactions is maintained by choosing appropriate read and write intersecting quorums and by frequently updating these subsets. Even though this technique is used to mitigate the effects of network partitioning, it is not fully aware of network partitions ahead of their occurrences. This technique also provides reasonable throughput even when there are many update transactions. However, it does not attempt to address real-time transactions and mobile hosts' power constraints.

#### 4.2 Effective replica allocation in ad hoc networks for improving data accessibility [11]

Hara [11] proposed three techniques, namely, static access frequency, dynamic access frequency and neighborhood and dynamic connectivity based grouping, to improve data accessibility in a MANET environment. These techniques make the following assumptions: (a) each data item and each mobile host is assigned a unique identifier; (b) every mobile host has finite memory space to store replicas; (c) there are no update transactions; and (d) the access frequency of each data item, which is the number of times a particular mobile host accesses that data item in a unit time interval, is known and does not change. The decision of which data items are to be replicated on which mobile hosts is based on the data items' access frequencies and such a decision is taken during a certain period of time, called the relocation period.

##### 4.2.1 Static access frequency (SAF)

Each mobile host arranges data items in the descending order of their access frequencies. The objective is to allocate a



mobile host with replicas of data items that are accessed most frequently from it. However, if there are mobile hosts with the same access characteristics, there would be redundancy in replica allocation between mobile hosts.

#### 4.2.2 Dynamic access frequency and neighborhood (DAFN)

In this technique, each mobile host preliminarily determines the allocation of replicas using SAF. It then eliminates replica duplication between all neighboring nodes in the following manner:

For all neighboring nodes (*node1* and *node2*) that are connected, the following procedure is repeated in the order of breadth-first search, starting from the host with the least identifier:

- If *node1* and *node2* have a data item redundant between them and *node1* holds the original data item and *node2* holds a replica of that data item, *node2* replaces this data item with another data item that has the next highest access frequency (AF) on *node2* and has not been replicated earlier.
- If both *node1* and *node2* hold replicas of the same data item, the node which has a lower access frequency of that data item replaces it with another data item that has the next highest AF on it.

This technique reduces replica redundancy compared with the SAF method, but it does not completely eliminate replica duplication among neighboring nodes as it only executes the elimination process from each mobile host by scanning the network once based on breadth-first search.

#### 4.2.3 Dynamic connectivity based grouping (DCG)

The DCG method tries to reduce duplicate replication in large groups of mobile hosts (or biconnected components) rather than removing duplication among neighboring hosts. A biconnected component is a maximum partial graph that remains connected even after the removal of an arbitrary node from the graph.

By dividing nodes into biconnected components, groups of mobile hosts are formed. These groups are not disconnected due to the failure of a single arbitrary host from the network, and hence such groups are highly stable. All bi-connected nodes in the network graph form different groups. Since groups of bi-connected nodes have high reliability in terms of network connection, replica redundancy in these groups can be removed. This is the objective of DCG.

A list of data items sorted in descending order of their access frequencies of a group is created. In the order of this list, each data item is allocated at a node from which the

**Table 1** Access frequencies of data items in individual mobile hosts and in groups

Data	Mobile host		Group G1
	M1	M2	
D1	0.97	0.88	0.98
D2	0.42	0.46	0.65
D3	0.45	0.37	0.55

access frequency of the data item is highest compared with other nodes in its group that have free memory space. After allocating replicas of all the data items, if there is still free memory space at some nodes in the group, replicas are allocated in descending order of access frequencies in those particular nodes until the memory space is full.

To illustrate DCG further, consider the following scenario where mobile hosts  $M_1$  and  $M_2$  form a bi-connected group  $G_1$ , having the following group access frequencies of three data items  $D_1$ ,  $D_2$  and  $D_3$  (Table 1).

In this particular scenario,  $D_1$  is a frequently accessed data item from both  $M_1$  and  $M_2$ . When the DCG technique is used, there is a possibility that  $D_1$  is not assigned to  $M_1$  and assigned only to  $M_2$ . This is because DCG tries to eliminate replica duplication without considering how high the access frequency of the replaced data item is. The objective of such elimination is to increase data availability, assuming that a request for  $D_1$  from  $M_1$  will be successful by forwarding the request to  $M_2$ . However, in a MANET environment where disconnection of mobile nodes is frequent, such a technique might be less efficient in terms of performance and reliability as  $M_2$  might not always be reachable from  $M_1$ . A suitable threshold value may be chosen so that the replacement occurs only if the access frequency of a data item is less than the threshold value.

All the above three techniques, SAF, DAFN and DCG, make the decision to replicate data during the relocation period based on the current network topology and the access frequencies of the mobile nodes, and the latter two also attempt to reduce any unnecessary redundancy in replica allocation. If the mobile nodes move to different locations over a period of time, the decision to replicate data items is taken based on their new locations in the next relocation period. However, these techniques do not address the issues caused by network partitioning, real-time transactions and the energy limitation of mobile nodes.

#### 4.3 Replica allocation with data update [12]

In [12], extensions to the above three techniques were proposed to improve data accessibility in an environment where

both periodic and aperiodic updates can occur. These extensions assume that every data item is broadcasted periodically. Each data item is updated only by the mobile host that has the original copy of the data item. Replicas of a data item become invalid if the nodes that contain these replicas cannot connect to the node that has its original copy. The proposed extensions are based on the access frequencies of data items (as those presented in [11]), as well as their *PT* values. A *PT* value of a data item represents the average number of requests that are issued for the data item before it is updated next. This is defined as the product of the access frequency of the data item and the time remaining until this data item is broadcasted next. A data item that has a higher *PT* value will then be replicated before a data item that has a lower *PT* value. This is done to ensure effective replication of data items considering the number of accesses prior to the next update. All these three extended techniques, E-SAF (extended SAF), E-DAFN (extended DAFN) and E-DCG (extended DCG), work the same way as SAF, DAFN and DCG except that the *PT* values, instead of the access frequencies, are used.

All of the three extended techniques assume that data updates occur periodically. In systems where there are aperiodic updates, the process of accessing a data item is changed as follows:

1. A request from a node to access a particular data item succeeds instantly if this node is the original node of this data item.
2. Otherwise, the node broadcasts a data search packet.
3. Every node that receives a data search packet inspects its database to check the availability of the data item and replies to the requested node.
  - a. If the received reply is from a node that contains the original copy of the requested data item and if the requested node contains a replica of this data item, it checks if an update has already occurred. If there are no updates, the requested node can access its replica directly; or else, it gets the data item from the original node.
  - b. If the reply is only from the nodes that contain the replica of the requested data item and if the requested node contains a replica of this data item, then the requested node can access it directly. Otherwise, it gets this data item from the nearest replicated node. This sometimes becomes a dirty read as it might not always get the most recently updated value.
  - c. If there is no reply from any of the nodes and if the requested node has a replica of that data item, it accesses the replica. If not, the request fails.

Using the replication techniques described above, the average number of requests issued for a data item  $D_j$  before it is

updated next can be computed by the following expression:

$$p_{ij} \int_0^{\infty} F_j(t + t_j) t \, dt$$

where  $p_{ij}$  is the number of times the data item  $D_j$  would be accessed by a node  $M_i$  in a unit time interval and  $F_j$  is the probability density function of the update intervals. Using this expression, [12] also proposed the following two extensions.

#### 4.3.1 Maximizing the number of successful access requests

The computed average number of requests issued for data item  $D_j$  is also a measure of the effectiveness of holding a replica of data item  $D_j$  whose update occurred before  $t_j$  units of time. This value is considered the new *PT* value and all the three extended techniques are further extended by using this new *PT* value in their algorithms. By using a higher *PT* value, the number of successful access requests is expected to increase.

#### 4.3.2 Reducing the rate of reading dirty items

The objective of this extension is to reduce the rate of dirty reads. If the replica for a data item  $D_j$  is held for  $t_1$  units of time, the total number of access requests issued within  $t_1$  is  $p_{ij} * t_1$ . Hence, the number of dirty read operations is the difference between  $p_{ij} * t_1$  and the average number of requests issued for a data item  $D_j$  before it is updated next (which is computed by the integral expression described above). A suitable value for  $t_1$  is chosen so as to reduce the number of dirty read transactions below a threshold value. This new *PT* value is then used in all of the extended methods.

As in [11], these techniques address the issues caused by host mobility as they take the decision to replicate data items every time during the relocation period based on the current topology of the network and the access frequencies of the mobile hosts. The proposed techniques also work efficiently in an environment where there are frequent updates. However, these techniques do not address the issues caused by network partitioning, real-time transactions and the energy limitations of the mobile nodes.

#### 4.4 Replica allocation for correlated data items [14]

Hara et al. [14] proposed an extension of the replication techniques described earlier in [11] to replicate data based on the “data priority” of data items that is based on the “correlation” among data items. Correlation between data items is the probability that clients collectively access a set of data items. Stronger the correlation, higher the probability that a

particular set of data items will be accessed together. Hara et al. [14] assume that clients access a set of correlated data items by submitting multiple access requests at the same time. Apart from the assumptions described in [11], Hara et al. [14] assume that the strength of correlation between the data items is known and does not change.

Replicas are relocated in a specific period called the relocation period as described in Sect. 4.2. In every relocation period, the replica allocation is determined based on the correlation between data items and the network topology at that moment. The replica allocation is determined based on the data priority of each data item as follows:

1. The access frequency of a data item  $D_j$  is calculated using the formula  $F_{ij} = \sum_{k=1}^n p_{i-jk}$  [14]. Here,  $p_{i-jk}$  gives the frequency (number of times) that a mobile host  $M_i$  accesses data items  $D_j$  and  $D_k$  together. If  $n$  is the total number of data items, given a constant  $x$  (the number of data items accessed together), there will be  $\lfloor n/x \rfloor$  access frequencies, considering the access of the data items being made together. Among the  $\lfloor n/x \rfloor$  possibilities, two combinations of data items that have the strongest correlation are chosen.
2. From the remaining data items (that were not chosen), the data item that has the strongest correlation with those data items that are already stored by  $M_i$  is chosen. This step is repeated until all the data items have been chosen.
3. The data items whose original copy is stored at  $M_i$  are assigned highest priority. Then, the data items chosen in Step 1 are assigned the next highest priority. All other data items are given a priority based on the order in which they were chosen in Step 2.

These extended techniques replicate data items based on the access relationship between them. They also address the issues caused by host mobility as the decision to replicate data items is taken periodically based on the current topology of the network. However, the techniques might yield the best results only when most of the data are accessed in correlation with each other. In addition, they do not address the issues caused by network partitioning, real-time transactions and energy limitation of mobile nodes.

#### 4.5 REDMAN: a decentralized middleware solution for cooperative replication in dense MANETS [1]

Given a resource (data item) and the number of copies of the data item required to ensure data availability across the network (called the degree of the replica), REDMAN (replication in dense MANETS) tries to allocate replicas of data items and maintains the number of replicas by reacting to any network topology changes that may occur with time.

However, it does not guarantee data consistency and targets applications that can tolerate stale or outdated data. Furthermore, since the technique considers dense MANETS, it excludes the possibility of network partitioning and merging. It also considers read-only data items, or, in the presence of updates, it does not guarantee data consistency. Nodes belonging to the dense MANET are identified by checking if their neighbor count exceeds a preset constant  $n$  and their neighbor density remains constant for a long period of time.

REDMAN distributes replicas in the network in order to ensure effective data availability. It also specifies a lightweight protocol for replica access, apart from providing a mechanism for maintaining the replica degree with changes in topology. The mechanisms for replica distribution, replica retrieval and replica maintenance are described in the following sections.

##### 4.5.1 Replica distribution

Each shared resource is associated with a metadata-based description that includes the replication degree of the resource. The replica manager has a shared resource table (SRT) that contains the shared resource and the degree of replication to be maintained. After populating the table, the manager appoints the owner of each data item as a delegate of the respective data items. Each delegate selects one of its neighbors randomly. It then forwards the data item to the chosen neighbor, and the data item is disseminated in this direction. Replicas of the same data item are stored at  $k$ -hop distance between each other. When a mobile host stores a copy of a data item, the replication degree of the data item is decreased by 1. The delegate continues to distribute the data item until the replication degree of the data item is reached.

##### 4.5.2 Replica retrieval [2]

In order to facilitate the access of replicas, REDMAN proposes a mechanism to propagate the information about replica placement (IRP) in the network. This scheme is called straight IRP dissemination (SID). Since propagating IRP to all devices in the network results in an overhead of communication, SID propagates IRP to only some mobile hosts. It coordinates with the replica distribution, and the IRP is propagated in the direction of replica distribution. Due to this, the IRP is stored in every mobile host between the delegate and replica servers of every data item. In order to access data items, clients send their access request in any direction. Bellavista et al. [2] observes that the probability of overlap in the direction of replica distribution and replica access is high. In addition, since mobile hosts between the delegate and replica servers of a data item store IRP, the data can

be accessed by the client when the access request crosses the direction of replica distribution. If an access request fails when propagated in a specific direction, clients choose a different direction to send the request.

In order to maintain the availability of IRP in the network with movement of mobile hosts, REDMAN also proposes a maintenance protocol for SID called SID reconstruction. When a mobile host  $S$  detects that any of its neighbors in the path of replica distribution has moved away, it broadcasts a reconstruction message to its neighbors. When the neighbors reply to the broadcast message,  $S$  chooses one of the neighbors to store the IRP. The IRP distribution is repeated by  $S$  when none of its neighbors respond to its reconstruction message.

#### 4.5.3 Replication degree maintenance [2]

Update dissemination for REDMAN has been presented in [2]. It tries to exploit the ad hoc collaborations between the mobile hosts to replicate in an efficient way, tolerating data inconsistency.

When a delegate foresees that it is exiting the dense MANET, it copies its shared data items to its neighbors. These neighbors communicate with the replica manager (which updates the SRT). If a delegate foresees its movement, it notifies the manager of its movement as well as the data items that it hosts. In the case that a delegate fails or moves out of the dense MANET without notifying the manager, a delegate is not chosen by the manager until it identifies the disconnection of the delegate through periodic beacon broadcasts between mobile hosts in the dense MANET.

REDMAN maintains the replication degree for each shared resource without guaranteeing absolute consistency. REDMAN simply reacts to any topology changes in the MANET and/or any changes in the network. A node, in order to replicate its resources when it moves into disconnection, notifies the managers whose region the node leaves and whose region it joins. A manager whose region the node is leaving chooses another node to replicate the resources of the outgoing node. Nodes may also advise the manager regarding the replication degree to be maintained for each resource.

REDMAN maintains the given set of replicas as needed by the application. However, it does not provide any mechanism to determine the optimal number of replicas required for each data item that will ensure data availability throughout the network. Furthermore, it does not provide any mechanism for replica synchronization or update propagation.

REDMAN does not take care of node power restrictions or transaction deadlines. It reacts to network partitioning by detecting changes in the neighborhood of servers, but is unable to know its occurrence ahead of time. Hence, this

technique is non-power-aware, non-real-time and non-partition-aware.

#### 4.6 Replica distribution method considering the location of mobile hosts in wireless ad hoc networks [27]

As the title indicates, [27] attempts to allocate and manage replicas considering the location of mobile hosts present in the network. The architecture considered by this technique consists of a peer-to-peer architecture where mobile nodes communicate with each other for information. The technique assumes the availability of position information of all mobile hosts at every other mobile node using a geographical positioning system like GPS, and that the probability of data access by a given node is defined by the distance between the node and the location where the data was generated. Furthermore, mobile hosts are assumed to access data relevant to their location. When a new data item is generated, the generating node propagates the data item to its neighbors. Distribution and replication of data objects as well as location based replication control are viewed as important factors influencing data accessibility. Under such conditions, the skip copy (SC) method described below is used for propagation of data items.

##### 4.6.1 Skip copy (SC) method

When a host generates a new data item, it broadcasts it to its neighbors in a packet that consists of a variable  $c$  initialized to 0, the current location of the host ( $P$ ), the ID of the mobile host, and the time of generation of the data. Upon receipt of this packet by any host,  $c$  is incremented and the data item is forwarded if the distance between the host and the original location of the data item ( $P$ ) is smaller than a preset parameter  $R$  called the replication range. Every node that receives the data item stores it if the condition  $(c \bmod S) = 0$  is satisfied, where  $S$  is a system wide parameter, called the skip parameter. The skip parameter is used to control replica redundancy and to ensure location based replica control (simply put, to make sure that replicas of the same data item are not located very close to each other). In the case that the node does not have free space available, it replaces the least frequently accessed data item, or the data item that was generated at the farthest location.

##### 4.6.2 Data access

When a requesting host does not have a copy of a required data item, it broadcasts a data access request for the data item to its neighbors. Such data access uses the location where the data item was generated as a search key. If a host that receives the access request has a copy of the data item being queried, it sends a copy of the data to the requesting host, or sends the



request to all its neighbors. In order to ensure that this does not result in a flooding of the request throughout the network, every node  $N$  forwards requests for data item  $D$  only once. The request is also forwarded only if the distance between  $N$  and the location where  $D$  was generated is less than the distance between the client that requests  $D$  and the location of the generation of  $D$ .

Before forwarding a request, every node appends its address to the message. This is done to store the route between the responding node and the client in the request message.

#### 4.6.3 Replica relocation

The replica relocation procedure is triggered after the broadcasting of an access request. Replicas of a data item are created at hosts that are at a distance  $S$  (hops) away from the location where the data item was generated. A minimum hop distance of  $S$  between hosts that have the replica of the same data item is ensured. When a responding host sends a copy of a data item, it sends a variable  $c$  initialized to 0. If any node in the route copies the data item, it increments  $c$ . When the requesting host receives the requested data item, it copies the data item if  $c \bmod S = 0$  and the distance between the nearest host that has a copy of the data item and the location where the data was generated is smaller than  $R$  (the replication range).

#### 4.6.4 Replica update

Each host maintains a timestamp list and a corresponding data item ID list. When a data item is updated, mobile hosts exchange the timestamps of the updated data item and determine the most updated copy of the data. The hosts then replace the old value of the data item with the new value. However, there is no guarantee that all nodes will get the most recent value of the data item as disconnections may occur frequently.

Tamori et al. [27] improve data accessibility by allocating replicas of data items that are distributed across the network and are far away from each other. While the skip parameter ensures that the replicas of the same data item are not too close to one another, the replication range ( $R$ ) ensures that the replicas are not separated by more than  $R$  hops. However, this technique neither attempts to conserve the power of mobile hosts, nor considers network factors like disconnection or partitioning as a factor to trigger replica allocation/relocation. Furthermore, the technique relies on the assumption that data access from a mobile host depends on the distance between the host and the location where the data item was generated. Therefore, in the absence of a positioning system, the replication feature is also disabled and, hence, cannot be used by applications not equipped with a positioning system.

## 5 Power-aware, non-real-time-aware and non-partition-aware techniques

Since nodes in a MANET are mobile, it may not be possible for them to charge their batteries when required. Due to the limited battery capability of mobile hosts, power is a much needed resource in MANETs that should be conserved as much as possible. Hence, a good replication technique should be able to reduce the power consumption of the mobile hosts in the network. Replication techniques that consider only the client and server power restrictions are referred to as power-aware, non-real-time-aware and non-partition-aware techniques.

### 5.1 Energy-efficient replication extended database state machine in MANET [24]

The database state machine (DSM) approach [25] uses a lazy master replication scheme in which one of the allocated replicas is considered the primary copy, and the node that hosts the primary copy is referred to as the master. Update transactions are always performed only at the master node, and they are propagated to other replicas, which are referred to as secondary copies. DSM reduces the power consumed by the mobile hosts by eliminating the need for atomic commit messages during the update propagation process. It implements an order mechanism in its broadcast scheme that ensures that each replica server receives and processes write transactions in the same sequence. Since the same order of update processing in the replica servers is ensured, DSM guarantees the correct ordering of update operations, while at the same time, conserves the power consumed by mobile hosts in this process.

Moon and Cho [24] proposed an energy efficient eager replication scheme, named E-DRM (eager replication extended database state machine), that focuses on achieving data consistency across the network while conserving energy by reducing the number of broadcast messages. The number of broadcasts is reduced by buffering all data to be transmitted over the network and broadcasting the data collected in the buffer periodically. The periodic cycle when the node flushes its buffer into the network is called the broadcast cycle. E-DRM basically extends the database state machine scheme; the architecture considered by it consists of small and large mobile hosts as defined in [9, 10] as follows:

1. SMHs (Small mobile hosts): These devices have limited memory, storage power and computing capabilities. They typically store only the query processing module because of such resource limitations.
2. LMHs (Large mobile hosts): These are computers that have higher memory, storage power and computing

capabilities than SMHs. They store the entire database system and respond to queries posed by the clients.

According to this technique, any node that needs to access or update data is a client. Clients submit transactions to any SMH. The SMH, after receiving the transaction, executes it, stores the results tentatively, and forwards the transaction to LMHs as soon as a broadcast cycle starts. This is done to reduce the power consumed by SMHs when broadcasting each update transaction to LMHs individually. LMHs then certify the tentative transactions as being correct through message broadcasting. LMHs may also make use of a single broadcast cycle to certify transactions to conserve power. Every LMH maintains an update information table (U-TBL) to validate global serializability. The update information table registers the data items that are updated during every broadcast cycle. After the SMH sends a commit request to an LMH, the LMH processes the transaction locally and broadcasts the information, including the readset and the writeset of the transaction to all LMHs for certifying the tentative transactions (executed by SMHs). Unlike conventional eager replication schemes, E-DRM broadcasts the information after the execution of read operations by grouping the results together and sending them at the beginning of the nearest (in time) broadcast cycle. A transaction, thus, goes through the following four phases [24]:

- *Request commit phase*: The SMH sends the tentative transaction information to an LMH, and the transaction enters its executing state.
- *Information broadcasting phase*: The LMH processes the read operations of the tentative transaction. It broadcasts the transaction information which includes its read and write sets and the write operations in the transaction.
- *Certification test phase*: LMHs that receive the broadcast from the previous stage commit the transaction and certifies it as correct. The commit operation resolves all conflicts involved with the current transaction.
- *Data broadcasting phase*: The LMH broadcasts the data updated by the tentative transaction, and sends the result of the transaction to the SMH. The SMH then sends the result (commit/abort) to the client.

The proposed technique attempts to reduce the power consumed by LMHs and SMHs by transmitting only at the beginning of the nearest broadcast cycle. The technique, however, does not specify a mechanism to allocate or synchronize replicas, except when a transaction is certified as being false and does not attempt to cope with network partitioning or take into account real-time transactions. E-DRAM is a server and client power-aware replication technique since it tries to reduce the number of broadcasts in both types of mobile hosts. However, the power savings at the server will

be appreciable only when the incoming queries are evenly distributed, and arrive in batches in the same interval of time.

## 5.2 Probabilistic quorum systems [22]

Classical ways of solving replica inconsistency problems, such as ROWA (read one write all) and atomic broadcast that disseminate update information to all replicas involve large communication overheads [22]. Luo et al. [22] proposed a set of protocols (PAN — probabilistic quorum system for ad hoc networks) that use a gossip-based multicast protocol to probabilistically disseminate updates in a quorum system to achieve high reliability even when there are large concurrent update and query transactions.

A quorum system is a set of quorums, each of which consists of a subset of server replicas. A quorum can be a write quorum accessed by an update transaction or a read quorum accessed by a read/query transaction. Consistency is ensured by propagating updates to the intersection of read and write quorums so that there are no dirty read transactions. Such an approach is termed “Write Many Read Many.” Probabilistic quorum systems introduce certain properties for the intersection of these quorums so that the most recent value of the data item is accessed by querying individual quorums.

Luo et al. [22] assumes a set of  $N$  nodes, a subset of which is called the storage set (STS), which holds shared data in a replicated environment. This subset of nodes is predetermined based on the power level of each server. The nodes present in STS may vary from time to time due to frequent network disconnections. The objective of this technique is to maximize the probability of fetching the most recent value of the data item required by a read transaction while minimizing the average network load per unit time, considering the update and query arrival rates. A client that needs services from a server sends its request to a random server in the STS. This server, also called the agent for that particular client, performs the following protocol.

In the case of an update transaction, the agent propagates the update information within the STS with the aid of other servers. Each server contains the following data structures:

- *midList*: This is the message ID of the most recent update.
- *Buffer*: This is a temporary storage location of the data updates.
- *View*: This is used to manage the membership of the STS. It has the following fields:
  - *AView*: The IDs of the known servers, the routing/location information of which is known.
  - *PView*: The IDs of the known servers, the routing/location information of which is not known.
  - *RView*: The IDs of the servers that are about to leave the STS.

Each receiver receiving the update message stores the updates temporarily in the buffer and updates the midList. The update process is executed every  $t$  seconds to disseminate the message stored in the buffer. Any server that receives this update message using a gossip protocol forwards it to randomly chosen receivers based on the current membership of the STS (which is stored in the View). Higher priorities are assigned to servers that are closer (by number of hops) to this server.

The system assumes fewer read transactions than update transactions and, hence, the size of read quorums is smaller than that of write quorums. A server (agent) that receives a read/query request from a client sends it immediately to other servers, indicating the version number of the corresponding data item. The servers in the read quorum of this message reply with a version of the data item they possess. The agent initiates the update process if it receives a new version of that data item from other read quorum servers.

Replica consistency is ensured despite frequent network disconnections by employing an asymmetric probabilistic quorum system that uses a gossip-based data dissemination technique. This technique chooses the nearest server to forward an update transaction in order to reduce network load. The servers in the storage set (STS) are chosen based on their power levels. This technique, however, is not aware of network partitioning. Furthermore, this technique proposes only a replica update strategy and does not propose methods to allocate or access replicas dynamically. This technique does not take into account transaction deadlines, and, thus, is non-real-time-aware.

## 6 Power-aware, real-time-aware and non-partition-aware techniques

Power on both clients and servers in a MANET is limited, as discussed in Sect. 2. Replication techniques that address both server and client power restrictions in addition to real-time constraints, but do not address the issue of network partitioning are called power-aware, real-time-aware and non-partition-aware techniques. To reduce power consumption, a good replication technique should reduce communication messages between various mobile hosts. It should also distribute and balance the work load of the servers in order to prevent the situation in which only a few servers execute all the transaction requests and run out of power, while the rest of the servers remain idle. Furthermore, to address the requirements of real-time applications, the replication technique should improve data availability for transaction execution in a timely manner. We have identified one technique, namely [26], that belongs in this classification. This technique is described in the following subsection.

### 6.1 Group based replication technique [26]

Shaheen and Gruenwald [26] proposed an energy efficient data replication technique for mobile ad hoc network databases. This technique is an improvement over the group-based replication for mobile ad hoc databases (GBRMAD) proposed in [9]. This section first describes the GBRMAD technique, and then discusses the improvements proposed for the technique in [26].

GBRMAD uses an MANET system model as proposed in [9, 10] which consists of two different types of mobile hosts, small mobile hosts (SMHs) and large mobile hosts (LMHs). These two types have been described earlier in Sect. 5.1.

While the SMHs function as clients in this architecture, the LMHs function as servers. Initially, a static allocation of data is assumed on LMHs and each SMH is assigned to a corresponding LMH. The mobile hosts are divided into groups, and each LMH belongs to a single group. Groups are defined in such a way that there is an equal distribution of servers having various power levels in all groups. If a data item  $X$  is primary to a group, all servers in the group will have a primary copy of data item  $X$ . Secondary copies of data item  $X$  are always allocated to groups other than the primary group. The read access ratio of each group  $G_i$  is defined as  $RRi = \frac{A_i}{\sum_{r=1}^m A_r}$  where  $A_i$  is the number of accesses in  $G_i$  and  $m$  is the number of groups in the system. The groups that have a higher read access ratio are assigned more servers. Each data item is stored as a tuple consisting of a value and a timestamp, and the one with a higher timestamp is selected for read.

If a server belonging to the primary group of data item  $X$  wants to read  $X$ , it issues a lock request to all servers in the group. The server then waits for a time period  $t_{\text{Timeout}}$  for the other servers to respond. In the event that a transaction needs the most recent value of the data it accesses, it requires at least  $\lceil N_{SGi}/2 \rceil$  locks (where  $N_{SGi}$  is the number of servers in group  $SG_i$ ) to successfully access the data.

If the required number of locks is obtained, data is from the nearest  $\lceil N_{SGi}/2 \rceil$  servers are read, or else the transaction is aborted. Data are read from more than one server to be able to identify the most recent value of the data item and use it for the execution of the transaction. If the transaction does not require the most recent value of the data item, a lock is requested from a server that is closest to the requesting host. Data are then read from this server.

If the node that wants to read the data item  $X$  does not belong to the primary group of  $X$ , it checks for a secondary copy of  $X$  on the local host. If a secondary copy of  $X$  exists, the node accesses this copy locally; otherwise it requests locks on the primary servers of  $X$  and accesses  $X$  from the primary group. The node also creates a secondary copy on

the local host if the number of reads processed for  $X$  is more than the number of writes.

When an update takes place in the primary group of data item  $X$ , the primary servers of  $X$  initiate the deletion of the secondary copies of  $X$  in other groups. In case the primary group of  $X$  is disconnected from a secondary copy server, the secondary copy server deletes the data item  $X$ . This is done to ensure that no stale secondary copies of a data item exist in the system.

When an update transaction for data item  $X$  is generated in a secondary copy server of  $X$ , the server requests exclusive locks on all the servers in the primary group of  $X$ . The server then waits for a time period  $t_{\text{Timeout}}$  for the servers to respond. Once the locks are obtained, if the transaction is a firm transaction, the nearest  $\lfloor (N_{SGi}/2) + 1 \rfloor$  servers are updated. If the transaction is soft, the servers are arranged in the descending order of power and the first  $\lfloor (N_{SGi}/2) + 1 \rfloor$  servers are updated.

Shaheen and Gruenwald [26] argue that in a system where the number of reads is assumed to be more than the number of writes, reducing the cost of reads will increase the system throughput even if the cost of updates increases. Hence, [26] uses an adaptive weighted majority voting to ensure concurrency of transactions in the system. The adaptive weighted majority voting aims at reducing the number of locks required for read operations, while incurring an increase in the cost of writes. In order to require a small number of locks for reads, and, at the same time in order to ensure that no write operation will access a data item that is currently being read, the adaptive weighted majority voting technique requires the following number of locks for read and write transactions on that data item:

$$\begin{aligned} & \# \text{ of locks required for read operation on data item } X \\ &= \left\lceil \frac{\text{Total \# of Write Requests for } X \text{ in the system}}{\text{Total \# of Write Requests for } X \text{ in the system} + \text{Total \# of Read Requests for } X \text{ in the system}} * 100 \right\rceil \\ & \# \text{ of locks required for write operation on data item } X \\ &= \left\lceil \left[ \left( \frac{\text{Total \# of Read Requests for } X \text{ in the system}}{\text{Total \# of Write Requests for } X \text{ in the system} + \text{Total \# of Read Requests for } X \text{ in the system}} * 100 \right) + 1 \right] \right\rceil \end{aligned}$$

Since the number of writes in the system is assumed to be small, the number of read locks required by reads will be small. This is because from the formulae above, the number of read locks is proportional to the number of write requests for that data item. This approach also entails an increase in the cost of write operations. The correctness of this concurrency mechanism is obvious from the above expressions, since when a data item is locked for read operations, it is not

possible for a write operation to acquire the required number of locks to succeed.

Hence, using an adaptive weighted majority voting scheme, [26] proposes an energy efficient data access method that ensures concurrency. In addition, since the updates are sent to all primary servers, the consistency of replicas is also maintained.

## 7 Non-power-aware, non-real-time-aware and partition-aware techniques

Network partitioning decreases data availability of MANET databases to a large extent. A good replication technique should be designed to maintain data availability at a desired level even during frequent disconnections. The existing replication techniques that do not address the issues of power consumption and real-time transactions, but do address network partitioning are called non-power-aware, non-real-time-aware and partition-aware techniques.

### 7.1 Service coverage in partitionable MANETS [28]

Wang and Li [28] proposed a set of run time algorithms to ensure the availability of centralized services to all mobile nodes in networks that are frequently partitioned. It assumes that the partitioning is caused only due to mobility of nodes and not due to lack of transmission power of nodes. The disconnections considered by this technique are caused by group-based node movement. The authors observe that in practice, mobile users have common tasks to perform and similar destinations and, hence, correlated mobility patterns. Due to such correlated movement, large-scale network-wide topology changes occur. This gives rise to disconnected partitions whose connectivity cannot be restored by topology control or by adjusting the node's transmission power or trajectory.

Mobile users in such frequently partitioned networks exhibit correlated mobility patterns, that is, their mobility patterns tend to be based on the group to which they belong [28]. A survey of the different mobility models used in ad hoc networks can be found in [4]. The authors define the reference velocity group mobility model (RVGM) and use it to identify the various mobility groups in the network and their characteristics.

Once the mobility groups are identified, a server can then calculate the time at which the services present on it will have to be replicated using the velocities at which the groups move away from each other [28]. Each server tries to replicate all its services on nodes in other partitions and delay such a replication process as much as possible. At the time of replication, the replicating node selects a node in the other



(future) partitions that is farthest away from its group in terms of distance with respect to its position. This is because the farthest node will be the last node to cross the server's group, after which both the groups will belong to separate partitions of the network.

At a certain instant of time, multiple servers (several child/replica servers and a parent server) in the system may store the same data item. Hence, client first has to decide the best server to which it can send its queries. Since each partition group will have a child server or a parent server based on the above described partition-aware replication, the task of the client is reduced to the problem of finding nodes of the same group. A fully distributed grouping algorithm to identify such nodes is described below.

When network partitioning occurs, clients may get disconnected from servers with which they were connected previously. Hence, every client selects a server that is strongly connected to it. Each node constructs its own stability group, which is defined as the group of nodes with which it has reliable connectivity. Then each client selects its server from its stability group. In the case a client identifies more than one server where it could send its queries, it selects the server with the most similar velocity. This is because nodes in the same mobility groups tend to share stronger and longer lasting links between each other. Choosing the best server periodically allows clients to access services or data from the most stably connected server at that time. At the time of replication, the server verifies if the client has changed its preferred server. If it has, the server then replicates its service on the client.

This algorithm guarantees the replication of services available on the server at the time of partitioning so that the availability of resources is guaranteed in the network [28]. In addition, this work also proposed the two following techniques in order to improve service efficiency, that is, to minimize service costs, so that the number of replicas is kept at a minimum while guaranteeing the availability of network-wide service.

- *Server: Service termination:* Every group requires only one server to provide services of one kind. However, if there is more than one redundant server that provides a service, the server with the highest ID will continue to serve and the other servers relinquish their capability as servers. The terminated servers cannot be the parent or the child of the highest ID server since such a replication is done only when different mobility groups are to be served by the replicas.
- *Client: Service replication decline:* A client must decline the offer to replicate the services of a server if it finds another server that offers the same services to the nodes of its group; therefore, duplication of servers is reduced.

Wang and Li [28] use a sequential clustering algorithm, assuming a system that is based on the reference velocity group mobility model to identify network partitions and disconnection of mobile nodes. Each partition is then provided with a parent or a child server to offer services. However, this technique addresses neither the issue of power limitation of mobile hosts nor that of real-time transactions.

## 7.2 Replication Decision Algorithm Based on Link Evaluation Services in MANET [15]

Hauspie et al. [15] proposed a new metric for evaluating link robustness that is used to detect network partitions without using the services of a GPS. According to this technique, the decision to replicate data items is taken not only at the time of detecting a network partition, but also during the time when the condition of the wireless connections worsen in terms of reliability, bandwidth and delay. This is because in high density networks, the connection is reliable only as long as the server is near the client as they would be separated by fewer hops [15]. In such a case, replicating a data service on a host that is closer to the client enhances the chances of the client being able to access the data on the server.

This partition detection technique relies on two fundamental ideas:

1. If a path has many essential nodes (the path does not connect its end points if these nodes fail), it is unreliable or weak.
2. If two nodes are far from each other, routing messages between them might generate more duplicate packets, which will result in decreased available bandwidth and increased delay.

This technique first determines a set of  $k$ -sub-optimal paths between two given nodes,  $v$  and  $w$ , represented by  $SOP_k(v, w)$  with  $k > 0$ . A path  $P$  is a  $k$ -sub-optimal path if and only if

- Path  $P$  is free of loops; and
- Number of hops in  $P$  (denoted by  $|P|$ )  $< d(v, w) + k$ , where  $d(v, w)$  is the least possible distance between  $v$  and  $w$ .

Such a set lists all the links between nodes  $v$  and  $w$  that are loop free and have the number of hops between them less than a threshold.

It then determines a subset of this  $k$ -sub-optimal-path set that contains only disjoint paths. Such a subset is denoted by  $DSP_k(v, w)$ . The Robustness of the path between nodes  $v$

and  $w$  is given by

$$R_k(v, w) = \max_{C \in DSP_k(v, w)} \left\{ \sum_{p \in C} \frac{1}{|p|} \right\}$$

This formula for robustness of a link considers the number of disjoint paths for reliability and, at the same time, gives a higher weight to shorter paths since the values of the elements of the summation are inversely proportional to the length of the path. This is a theoretical approach to computing the robustness of the link between two nodes. In addition to this approach, the following three broadcast-based techniques were proposed as a practical approach to computing the robustness of the link connecting two nodes. In all these three techniques, it is the client that computes the robustness of the link with which it connects to the server. For the purposes of the discussion below, let  $v$  be the client trying to evaluate its link to the server  $w$ , and  $u$  every host that receives broadcast packets from  $w$ .

### 7.2.1 Efficient broadcast based protocol (mode 0)

In the mode 0 of broadcasting, the client ( $v$ ) initially requests the server ( $w$ ) for link evaluation using a routed message. The server then replies with a TTL (Time to Live) of  $d(v, w) + k$  using a flooding protocol. Every node that forwards this message appends its node ID in the message to determine the actual path of the message. The client, after the receipt of a TTL message, stores the path the message has traveled if it is disjoint from all other stored paths. After a certain previously set time interval, the client determines  $R_k(v, w)$ .

### 7.2.2 Stormy broadcast based protocol (mode 1)

The only difference between this approach and the mode 0 approach is that the server ( $w$ ), after receiving a link evaluation request from the client ( $v$ ), sends a list of sequence numbers to all its neighboring nodes. Each neighboring node ( $u$ ), on receipt of the request, chooses a random sequence number from the received list and, in turn, sends a link evaluation request with a TTL of  $d(v, w) + k$ , using a flooding protocol. Each node forwarding the message appends its ID to the message in order to facilitate the construction of the path traversed by the message. The client then compares the received path with its stored path set to determine  $R_k(v, w)$  as in mode 0. This method generates more number of disjoint paths since the link evaluation packet will traverse through most combinations of the route; however, it also incurs heavy network traffic, since the network is flooded with packets throughout the process.

### 7.2.3 Directed response based protocol (mode 2)

One downside of the mode 1 technique is that almost the entire network is flooded by packet exchanges. To prevent this flooding and to restrict the neighborhood/zone, through which the link evaluation is performed, the server ( $w$ ), after receiving a link evaluation request from the client ( $v$ ), sends a list of sequence numbers to all its neighboring nodes. Every node that receives this message ( $u$ ) forwards it only if  $|p(w, u)| + d(u, v) \leq d(v, w) + k$  and if it has not already forwarded a packet with the same sequence number. This condition is used to make sure that the link analysis is made only using routes with the number of hops less than or equal to that of a  $k$ -sub-optimal. It is imposed to make sure that the evaluation of the link between two mobile hosts is done using paths with lengths that are at most equal to the length (in hops) of a  $k$ -sub-optimal path. This technique was shown to be the best of all the three techniques by means of simulation.

These techniques predict network partitions and disconnections of mobile hosts by using a multiple disjoint path set that is both reliable, due to multiple paths, and efficient, as it gives priority to shorter paths over longer paths. Replication is triggered after the detection of network partitions as well as when the system determines that the connection is bad in terms of reliability, bandwidth and delay. However, this technique is neither power-aware nor real-time-aware.

## 7.3 Stability of radio links [13]

Hara [12] proposed three techniques, DAFN-S1, DAFN-S2 and DCG-S1, to enhance those proposed in [11] (discussed in Sect. 4.1) by addressing the issues caused by network partitioning. The objective of these techniques is to eliminate duplicate replicas of data items between two mobile hosts only when the wireless link between them is stable. There is no use in removing the duplicate replicas of data items in two mobile hosts that are connected by unstable radio links as the replicated data items cannot be shared by these two mobile hosts once they are disconnected [12]. The proposed techniques assume prior knowledge of the speed and direction of movement of all mobile hosts. The time at which two mobile hosts would disconnect can then be estimated by studying their movements. This disconnection time between two mobile hosts,  $M_i$  and  $M_j$ , is represented as  $T_{ij}$ . The stability of the wireless link between  $M_i$  and  $M_j$  is denoted by  $B_{ij}$  and, if  $T$  is the relocation period (as defined by Sect. 4.1),  $B_{ij}$  is given by

$$B_{ij} = 1 \quad \text{when } T_{ij} \geq T,$$

$$B_{ij} = \frac{T_{ij}}{T} \quad \text{otherwise}$$

### 7.3.1 DAFN-S1 (DAFN — stability of radio links: 1)

DAFN-S1 works the same ways as DAFN (described in Sect. 4.1) except that it takes into account the stability of wireless links while eliminating the duplication of replicated data items in any two neighboring mobile hosts. For each neighboring host  $M_j$  of a particular mobile host  $M_i$ , the stability of the associated radio link  $B_{ij}$  is calculated. The decision to eliminate redundancy of replicated data items between  $M_i$  and  $M_j$  is taken only when the value of  $B_{ij}$  is more than a certain threshold value. A higher value of  $B_{ij}$  means that the link connecting  $M_i$  and  $M_j$  is more stable and, hence, data can be shared successfully between them.

### 7.3.2 DAFN-S2 (DAFN — stability of radio links: 2)

DAFN-S2 uses a different strategy to evaluate the stability of wireless links to eliminate duplication of replicas. First, a host, the replica of which is to be replaced, is chosen using the same technique as in DAFN-S1 [13]. Let the access frequency of this data item on this host be  $p$ . The probability that this host would access the same data item even after disconnection is defined as  $((1 - B_{ij}) * p)$ . The replica is replaced only if this value is more than a certain threshold value  $x$ . A replica with a lower value of  $((1 - B_{ij}) * p)$  is preferably removed compared to one with a higher value as it implies a less probability for that mobile host to access the data item after disconnection.

### 7.3.3 DCG-S1 (DCG — stability of radio links: 1)

This technique considers two mobile hosts,  $M_i$  and  $M_j$ , to be *connected* only if  $B_{ij}$  is greater than a certain threshold value. Different *bi-connected* mobile hosts of the network then form different groups. Replica duplication in each group is eliminated as in the DCG method [11]. These groups are formed considering the fact that two nodes are connected only if the link between them is stable.

The three techniques, DAFN-S1, DAFN-S2 and DCG-S1, are aware of network partitions as they consider the stability of the radio links that connect the mobile hosts. Frequent disconnection and frequently changing network topology are taken care of, as the decision to replicate data items is made once in every relocation period. This technique, however, is not aware of real-time transactions and does not consider the energy limitations of mobile hosts.

## 7.4 Exploring group mobility for replica allocation in a mobile environment [18]

Group Mobility refers to the movement of several mobile hosts together. Huang et al. [18] proposed a replication method (DRAM) to improve data accessibility in a MANET

that allocates replicas, considering group mobility. In DRAM, every node first exchanges its motion behavior with some neighbors. This information exchange is limited by a predetermined parameter. Then a decentralized clustering algorithm was proposed to cluster mobile nodes with similar motion behavior into mobility groups. Clusters that are likely to be connected are merged into an allocation unit to share the data items stored by each other. Data items are then replicated according to the resulting allocation units. The technique assumes updates to be periodic in nature. DRAM maintains groups in an *adaptive* fashion which keeps the amount of information broadcast as small as possible to reduce network traffic.

Since network connectivity in MANET may change frequently, DRAM is executed periodically, once in every relocation period. Each mobile host performs two phases during the relocation period, Allocation unit construction phase and replica allocation phase. Allocation units are formed in the allocation unit construction phase. An allocation unit is defined as a set of mobile nodes which share their stored data and store no duplicate data unless all data items have been allocated in the unit [18]. Allocation units are formed when any mobile node broadcasts an information message that contains the node ID, the current time stamp and movement parameters according to the reference point group mobility model (RPGM) described in [17]. The mobile nodes are classified into groups by the lowest-ID clustering algorithm proposed in [7]. A broadcast zone is defined for each node as the set of nodes that are within a predefined distance TTL from the node. Each node in its broadcast zone that has the lowest node identifier is then chosen as the master of the broadcast zone. Zone masters then cluster member nodes having the same motion behavior. First, all nodes are clustered based on their motion and stationary timestamps, and they are then refined by considering their motion vectors. Each member then sends status messages to the cluster master and, when it is unable to reach the master, enters the initial state and joins another group. The master also removes nodes that do not send status messages from the cluster.

During the replica allocation phase, data items are allotted to different mobile hosts based on their access frequencies and the allocation units that were formed in the previous phase. The employed replication algorithm is as follows.

The allocation weight of data item  $D_j$  in allocation unit  $C_x$  in timestamp  $k$  (denoted as  $w_j^x(k)$ ) is the expected number of data accesses from all mobile nodes in  $C_x$  before the next update of  $D_j$ .  $w_j^x(k)$  is obtained as follows [18]:

$$w_j^x(k) = f_j^x * (U_j - k), \quad \text{where } f_j^x = \sum_{M_i \in C_x} f_{ij}$$

where  $f_{ij}$  is the access frequency of a data item  $D_j$  on a mobile host  $M_i$  such that it is the highest access frequency of  $D_j$  among the access frequencies of  $D_j$  in all other mobile

hosts that have free storage space.  $U_j$  is the time stamp of the next update of  $D_j$ , which is determined using the periodicity of updates on  $D_j$ .

Each unit master then allocates the data item with the largest weight to the mobile host where the access frequency is the largest until all the replicas have been allocated.

This technique uses the RPGM model [17] to identify network partitions. It assumes that there is a similarity between the movements of some of the nodes; however, if there is a large number of nodes that have diverse movement patterns, the groups formed may be small in size and very large in number. In addition, if random motion of nodes occurs, the group joining and leaving operation may become an overhead in the system. The proposed technique also does not take into account the real-time transaction requirements of applications and also does not consider the power constraints of mobile hosts.

## 8 Power-aware, non-real-time-aware and partition-aware techniques

A data replication technique for MANET databases that addresses both the issues of power limitation of mobile hosts and network partitioning, but does not consider real-time transaction requirements, is called a power-aware, non-real-time-aware and partition-aware technique. The following subsection describes one such technique that we have identified.

### 8.1 Predictive data replication [5]

Chen and Nahrstedt [5] proposed a distributed data lookup algorithm to address the issue of identification of data availability in MANET and a predictive data replication algorithm. This technique uses the group-based data accessibility scheme. In such a scheme, a set of mobile nodes forms a separate group and the nodes within this group collectively host a set of data items that are available for data access to all the other nodes of the group, while reducing data redundancy within that group. This approach is similar to that of DCG [11], except that the DCG defines biconnected groups of servers for the purpose of data replication. In addition, data access in DCG is not group based.

Data availability information is exchanged using a peer-to-peer data advertising protocol where each node periodically broadcasts an advertising message called “ad” to other peer nodes within a group. These ad messages contain a sequence number, the sender’s address, the resources (node’s free space, power available, processor idle percentage, etc.) and the data items available at the sender.

Each node, on receiving an ad message from other nodes, updates a local data availability lookup table for its own use.

Nodes, after gaining knowledge of data availability of other nodes, update their local replicas to avoid redundancy of data. Every data item is associated with a timestamp which is used in deleting the older version of the data item when a newer version is available. To address scalability issues concerning the size of the ad messages, a differential ad message that contains the difference of the ad contents from the previous message is broadcast more frequently, while complete ad messages are broadcast less frequently.

This technique uses the services of a location-based routing protocol [3,21,23] to obtain the geometric position, movement pattern and transmission range information of a node. It assumes that each node follows a predictable movement pattern like piece-wise linear motion, or a non-linear curve motion. The mobility of the nodes and frequent disconnections that cause network partitioning are predicted based on the current position of nodes and their patterns of movement. Data are replicated in appropriate partitions before the actual partitioning occurs so as to improve data accessibility.

Once the occurrence of a network partition is predicted, data on those nodes that may be rendered inaccessible due to network partitioning are replicated so that the resulting partitions will be able to access data belonging to nodes in other partitions. The selection of a candidate node is based on the nodes’ capabilities (free hosting space, remaining energy, processor idle percentage). In the case of a hosting space limitation, more frequently accessed data (hot data) replaces the less frequently accessed data (cold data).

Since this technique predicts the occurrence of network partitioning ahead of time and considers the power constraints of mobile hosts, it is both partition and power-aware. However, this technique does not consider real-time requirements of applications.

## 9 Feature comparison

In this section, we compare the existing data replication techniques that were classified and reviewed in Sects. 4–8 using a framework composed of thirteen critical features as given below.

- *MANET architecture*: For which type(s) of MANET system architecture(s) is the replication technique developed? The system architecture determines how mobile hosts communicate with each other. There are four types of architectures.
  - *Centralized architecture*: All mobile hosts send their queries to a central server. This architecture suffers from the single entry point failure problem, where the entire system fails if the central server fails.
  - *Group-based common server architecture*: Mobile hosts are arranged into different groups based on



various parameters. Mobile hosts may belong to more than one group. Each group is associated with a central server and clients belonging to a group access data through the central server. The central servers of different groups may be able to communicate with each other to synchronize data as well as to propagate group membership information.

- *Decentralized group-based architecture*: Mobile hosts are arranged into different groups based on various parameters like location of hosts, application semantics (e.g., clients in one region may group around the nearest server that has the data of common interest to all of the clients), etc. Every mobile host belongs to only one group. Mobile hosts in a group can communicate only with mobile hosts that are in the same group. There is no central server in any group, and clients in the group may access data from any server in the group.
- *Decentralized architecture*: A mobile host can send its transactions to any of the mobile hosts that are within its communication range.
- *Client/server classification*: Does the replication technique classify mobile hosts into clients and servers?
- *Server power-aware*: Does the replication technique address server power limitations?
- *Client power-aware*: Does the replication technique address client power limitations? Techniques that do not differentiate mobile hosts as clients/servers can also be aware of the power limitations of mobile hosts in general.
- *Real-time-aware*: Does the replication technique support firm and soft real-time transactions?
- *Read-only assumptions*: Does the replication technique assume that all transactions are read-only transactions?
- *Replication period*: Is the decision to replicate data taken more than once? If yes, how often?
- *Partition-aware*: Does the replication technique predict network partitions before they occur and replicate data accordingly?
- *Routing protocol dependency*: Is the replication technique dependent on any particular routing protocol?
- *Replication level*: What is the level of replication used? Is it full or partial replication?
- *Parameters that are considered for the decision to replicate*:
  - *Radio link stability*: Does the replication technique consider the stability of the wireless links while deciding if a particular data item can be replicated in a mobile host?
  - *Access frequency*: This indicates the number of times a data item was accessed by a mobile host. Does the replication technique consider this for replication decision?

- *Communication cost*: Does the technique consider the communication cost (e.g. number of hops, distances between the communicating hosts, strength of link connection between communicating hosts, etc.) in the decision of how the replication should be made?

- *Use of GPS (or other alternative node positioning method)*: Does the replication technique need the services of a global positioning system or any other node positioning method?
- *Network partitions detection*: How does the technique detect network partitions while taking the decision to replicate?

Based on the above features, Tables 2 and 3 compare the existing data replication techniques reviewed in the previous sections.

## 10 Matching data replication techniques to applications and performance comparison model for manet replication techniques

### 10.1 Matching data replication techniques to applications

MANET databases can be deployed in a wide range of applications. Since the choice of a replication technique is based on the characteristics of the application and network architecture, no single replication technique is suitable for all applications. The following constraints have to be considered when choosing a particular replication technique:

- a. *Real-time applications*: Many MANET applications, like battlefield and disaster recovery, are time-critical applications. In such applications, transactions are generally restricted to deadlines within which they have to be executed. A real-time aware technique is most appropriate in such cases. Since there is more than one such technique, the best suited technique can be determined by using the following performance metrics:
  - *Percentage of firm real-time transactions missing their deadlines*. A good real-time aware replication technique should reduce the number of firm transaction aborts due to the expiry of their deadlines.
  - *Percentage of soft real-time transactions missing their deadlines*. A good real-time aware replication technique should reduce the number of soft transactions missing their deadlines.

Furthermore, in the event of having to arrive at a compromise between firm and soft real-time transactions, a good replication technique should be able to act in accordance with the application semantics.

- b. *Frequency of network disconnections and network partitions*: Network partitioning occurs frequently in most

**Table 2** Features of existing data replication techniques

	[20]	[26]	[11]	[27]	[5]	[28]	[15]
MANET architecture	Group-based common servers	Decentralized group-based	Decentralized group-based	Decentralized group-based	Decentralized	Decentralized	Decentralized
Client/server classification?	Yes	Yes	No	No	Yes (based on MH capability)	Yes (based on service)	Yes (based on service)
Server power-aware	No	Yes	No	No	Yes	No	No
Client power-aware	No	Yes	No	No	Yes	No	No
Real-time aware	No	Yes	No	No	No	No	No
Read-only assumptions	No	#read > #write	Yes	No	Read only	No	No
Partition-aware	No	No	No	No	Yes	Yes	Yes
Routing protocol dependency	No	No	No	No	Location-based routing protocol	No	Flooding and broadcast
Replication level	Full	Partial	Partial	Partial	Partial	Partial	Partial
Link stability consideration	No	No	No	No	No	No	Yes
Access frequency consideration	No	Yes	Yes	No	Yes	No	No
Communication cost consideration	No	No	No	No	No	No	Yes
Use GPS or other positioning system	No	Yes	No	Yes	Yes	Yes	No
Relocation period	No	No	Yes	Yes (after every request broadcast)	No	No	No
Partition detection	No	No	No	No	Based on location, direction, velocity and transmission range of servers	Based on location, direction and velocity of servers	Based on the robustness of radio links between mobile hosts

**Table 3** Features of existing data replication techniques

	[13]	[12]	[22]	[18]	[14]	[24]	[1]
MANET architecture	Decentralized group-based	Decentralized	Decentralized	Decentralized group-based	Decentralized	Decentralized	Decentralized
Client/Server classification?	No	No	Yes (based on data availability)	No	No	Yes (based on mobile host's capabilities)	Yes (based on services provided)
Server power-aware	No	No	No	No	No	Yes	No
Client power-aware	No	No	No	No	No	Yes	No
Real-time aware	No	No	Yes	No	No	No	No
Read-only assumptions	Read only	No	#read > #write	No, updates aperiodic considered	Read only	No	No
Partition-aware	Yes	No	No	Yes	No	No	No
Routing dependency	No	No	Gossip based multicast	No	No	No	No
Replication level	Partial	Partial	Partial	Partial	Partial	Partial	Partial
Link stability consideration	Yes	No	No	No	No	No	No
Access frequency consideration	Yes	Yes	No	Yes	Yes	No	Yes
Communication cost consideration	Yes	No	No	No	No	No	Yes
Use of GPS or other positioning system	No	No	No	No	No	No	No
Relocation Period	Yes (Periodically according to a pre-defined relocation period)	Yes (Periodically according to a pre-defined relocation period)	No	No	Yes (Periodically according to a pre-defined relocation period)	No	No
Partition detection	Based on current location, direction of mobile host	No	No	Yes (based on mobile host's movement)	No	No	No

MANET applications. However, in military and similar applications, network partitioning may occur more frequently than in other applications like a tourist guide [28]. A replication technique that is aware of network partitioning is most appropriate in those cases. Such a partition-aware technique should ensure the timely availability of data even under conditions of network partitioning. The effectiveness of such a replication technique can be evaluated using the following metrics:

- *Average transaction response time.* Frequent network disconnections and partitions causes unavailability of required data that reside on remote hosts. Hence, such partitioning may increase the average response time of servers due to the unavailability of a required participating server for the successful execution of a transaction. A good replication technique should minimize the average transaction response time by replicating ahead of disconnection/partitioning occurrences.
- *Average system throughput.* Due to frequent disconnections and partitioning, and the unavailability of required participating servers, deadlines of transactions may expire, causing them to be aborted. Hence, such disconnections may cause an overall decrease in the number of transactions that are executed successfully and, thus, reduce the average system throughput. A good replication technique should maximize the average number of transactions that are completed in a unit of time by replicating according to network partitioning and disconnection patterns.
- c. *Storage size and power availability constraints:* Some replication techniques assume unlimited memory space for data replication, and that all mobile hosts have the same storage capability and power availability. Some MANET applications simultaneously use handheld devices (for higher mobility) and laptops. When the data size is extremely large, handheld devices may not be able to store the required amount of data for replication. Moreover, handheld devices are largely restricted by their power availability. In such an environment, replication techniques that are power-aware and pose a restriction on memory usage are required. The effectiveness of such a replication technique can be evaluated using the following performance metrics:
  - *Average storage space used by the system.* A good replication technique should reduce the average storage space used on the servers.
  - *Average energy consumption by servers.* A good replication technique should work in such a way as to minimize the average energy consumption of all servers.
  - *Distribution of energy consumption by servers.* A good replication technique should ensure that there is

not much deviation between the energy consumption of any two servers if all servers have the same system capability; otherwise, the energy consumed by a server should be proportional to its system capability.

- *Average energy consumption by clients.* A good replication technique should work in such a way that reduces the average energy consumption of all clients.
- d. *Update transactions:* Many real-time MANET applications need frequent update of data, e.g. military applications, where the enemy positions and resources are frequently updated. In such a scenario, replication techniques that address both read and write transactions are more appropriate than those that address only read transactions. The effectiveness of a replication technique in the presence of update transactions can be evaluated using the *number of dirty data reads* and the *number of successful transactions*, apart from the *percentage of firm real-time transactions missing their deadlines*, *percentage of soft real-time transactions missing their deadlines*, *average transaction response time* and *average system throughput* parameters discussed above.
- e. *Availability of GPS (or any other alternative to identify the positions/locations of mobile hosts):* Many partition-aware techniques use GPS to identify the position of a mobile host to detect network partitions; however, the replication technique discussed in [15] does not require GPS for detecting network partitions. Instead, it evaluates the optimality of paths using the number of hops to detect the occurrence of network partitions (Sect. 7.2). Such a technique is best suited for an ad hoc environment without any fixed infrastructure, including GPS.

## 10.2 Performance Comparison Model

Different articles reviewed in the previous sections provide their own terms of evaluation of their proposed techniques. For example, [14] represents the performance of their proposed technique by measuring the effect of relocation period and radio communication range on data availability and network traffic, while [13] measures the effect of available memory size on network traffic and data accessibility. Most of these evaluation metrics and parameters used therein are specific to the technique, the performance of which is being measured, and, hence, they mostly do not allow for a performance comparison between the different techniques. This is a serious problem because in order to select a data replication technique for use in a real application among all existing techniques, users should be aware of not only the specific features of the techniques (as listed in Tables 2 and 3), but also their performance data.

To enable users to compare the performance of different replication techniques, since there is no MANET database



benchmark available, experiments should be conducted by varying a standard set of dynamic parameters in a wide range of values to study their effects on a set of performance metrics. These experiments should measure the various performance metrics which were described earlier in Sect. 10.1 and are summarized in Sect. 10.2.1 below. The dynamic parameters, the effects of which these experiments should measure, are listed in Sect. 10.2.2.

### 10.2.1 Performance metrics

- Percentage of firm real-time transactions missing their deadlines
- Percentage of soft real-time transactions missing their deadlines
- Average transaction response time
- Average system throughput
- Average storage space used by the system
- Average energy consumption by servers
- Distribution of energy consumption by servers
- Average energy consumption by clients.

### 10.2.2 Dynamic performance parameters

- Transaction inter-arrival rate. This will allow users to choose a replication technique suitable for their applications, basing their comparison on the transaction workload that their application is designed to handle.
- Access frequency. This will give users an idea of how well the replication technique can cope with changing/fixed data access patterns.
- Effect of firm/soft transaction ratio. This will help users judge replication techniques based on how well each technique will be able to handle different combinations of real-time transactions.
- Frequency of network disconnection and partitioning. This will allow users to compare the performance of different replication techniques under various network conditions.
- Available storage size and power level. This will allow users to understand the storage and power required by a replication technique for optimal performance.
- Read/write transaction ratio. This will help users judge replication techniques based on how well the system will be able to handle different patterns of read and write transactions.
- Number of nodes. This will allow users to compare the extent of scalability of the replication techniques.
- Velocity of nodes. This will help users judge the performance of replication techniques under different scenarios of node mobility.

## 11 Conclusions and future research directions

We presented a survey and classification of data replication techniques for MANET databases. Issues to be considered when developing a replication technique for such databases were identified. Existing replication techniques were compared based on how they addressed the identified issues. Criteria for selecting a replication technique for a particular application were established. From this survey, we can see that research in MANET data replication is still in its infancy. We describe below a number of *open* research problems.

None of the existing data replication techniques addresses all the MANET issues, that is, none of them is power-aware, partition-aware as well as real-time-aware. Development of a data replication technique that considers all the three issues would be an attractive topic for future research. The ultimate aim of such a technique would be to maximize the number of transactions meeting their deadline constraints while minimizing the energy consumption of all mobile hosts, and alleviating the effects of network partitioning by appropriate data replication.

Current partition-aware techniques detect network partitions caused only by the mobility of nodes; however, network partitioning may also occur due to loss of energy of nodes. Techniques that detect partitions based on direction, velocity and movement patterns of nodes cannot detect partitions due to loss of energy of nodes; hence, in addition to node mobility, it is important to also consider energy as a factor for network partitioning. The energy consumption of mobile hosts depends on their workload. Hence, in order to be able to predict the occurrence of network partitioning due to loss of power a replication technique should be able to estimate the workload of the servers. In addition, network partitions may reconnect after some time, causing data replicated across the partitions to be redundant; hence, the replication technique should be able to remove replica duplication to improve data accessibility. Such an algorithm would be feasible only by a system having the capability of identifying reconnected partitions.

All but one of the techniques surveyed in this paper were evaluated and tested by simulation. The next step is to prototype these techniques in a real world scenario. This would provide a better perspective on the resource restrictions in MANETs, apart from providing a realistic evaluation of the assumptions made and the performance of the surveyed replication techniques. In addition to prototyping the existing replication techniques, a performance comparison of the existing replication techniques should also be performed.

There is no standard architecture for MANET databases that defines how a mobile host communicates with other mobile hosts for data access. A survey of different MANET architectures and their performance is impending, which

would help choose the most suitable architecture for a particular application.

**Acknowledgements** This work was partially supported by National Science Foundation Grant No. IIS-0312746.

## References

- Bellavista, P., Corradi, A., Magistretti, E.: REDMAN: A decentralized middleware solution for cooperative replication in dense MANETs. In: International Conference on Pervasive Computing and Communications Workshops, pp. 158–162 (2005)
- Bellavista, P., Corradi, A., Magistretti, E.: Comparing and evaluating lightweight solutions for replica dissemination and retrieval in dense MANETs. In: IEEE International Symposium on Computers and Communications, pp. 43–50 (2005)
- Camp, T., Boleng, J., Williams, B., Wilcox, L., Navidi, W.: Performance comparison of two location based routing protocols for Ad Hoc networks. IEEE INFOCOM, pp. 1678–1687 (2002)
- Camp, T., Boleng, J., Davies, V.: A survey of mobility models for ad hoc network research. Wireless Communication and Mobile Computing: Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, pp. 483–502 (2002)
- Chen, K., Nahrstedt, K.: An integrated data lookup and replication scheme in mobile ad hoc networks. In: SPIE International Symposium on the Convergence of Information Technologies and Communications, pp. 1–8 (2001)
- Davidson, S.B., Garcia-Molina, H., Skeen, D.: Consistency in partitioned networks. ACM Comput. Surv. **17**(3), 341–370 (1985)
- Gerla, M., Tsai, J.: Multiclust, mobile, multimedia radio network. ACM Wireless Netw. **1**(3), 255–265 (1995)
- Gray, J., Helland, P., O’Neil, P., Shasha, D.: The dangers of replication and a solution. In: ACM SIGMOD International Conference on Management of Data, pp. 173–182 (1996)
- Gruenwald, L., Mohile, S.: Energy efficient data replication for MANET. A Technical Report, University of Oklahoma, 2000, <http://www.cs.ou.edu/~database/documents/mohile.pdf>, accessed November 2006
- Fife, L., Gruenwald, L.: Research issues for data communication in mobile ad hoc network database systems. ACM SIGMOD RECORD, pp. 42–47 (2003)
- Hara, T.: Effective replica allocation in ad hoc networks for improving data accessibility. IEEE INFOCOM, pp. 1568–1576 (2001)
- Hara, T.: Replica allocation methods in ad hoc networks with data update. ACM Kluwer J. Mobile Netw. Appl. **8**(4), 343–354 (2003)
- Hara, T., Loh, Y.H., Nishio, S.: Data replication methods based on the stability of radio links in ad hoc networks. In: The 14th International Workshop on Database and Expert Systems Applications, pp. 969–973 (2003)
- Hara, T., Murakami, N., Nishio, S.: Replica allocation for correlated data items in ad hoc sensor networks. ACM SIGMOD RECORD, pp. 38–43 (2004)
- Hauspie, M., Simplot, D., Carle, J.: Replication decision algorithm based on link evaluation services in MANET. CNRS UPRESA 8022 — LIFL University Lille (2002)
- Helal, A.A., Heddaya, A.A., Bhargava, B.B.: Replication Techniques in Distributed Systems. Kluwer, Dordrecht (1986)
- Hong, X., Gerla, M., Pei, G., Chiang, C.: A group mobility model for ad hoc wireless networks. ACM/IEEE MSWIM, pp. 53–60 (1999)
- Huang, J.-L., Chen, M.-S., Peng, W.-C.: Exploring group mobility for replica data allocation in a mobile environment. In: 12th ACM International Conference on Information and Knowledge Management (CIKM-03), pp. 161–168, November 3–8 (2003)
- IETF MANET Home Page, <http://www.ietf.org/html.charters/manet-charter.html>, accessed July 2004
- Karumanchi, G., Muralidharan, S., Prakash, R.: Information dissemination in partitionable mobile ad hoc networks. In: IEEE Symposium on Reliable Distributed Systems, pp. 4–13 (1999)
- Ko, Y., Vaidya, N.H.: Location-aided routing (LAR) mobile ad hoc networks. In: ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM), pp. 66–75 (1998)
- Luo, J., Hubaux, J.P., Eugster, P.T.: PAN: Providing reliable storage in mobile ad hoc networks with probabilistic quorum systems. MobiHoc, pp. 1–12 (2003)
- Mauve, M., Widmer, J., Hartenstein, H.: A survey on position-based routing in mobile ad hoc networks. IEEE Netw. **15**(6), 30–39 (2001)
- Moon, A., Cho, H.: Energy efficient replication extended database state machine in mobile ad hoc network. In: IADIS International Conference on Applied Computing, pp. 224–228, (2004). <http://www.cs.ou.edu/~database/documents/moon04.pdf>
- Pacitti, E., Minet, P., Simon, E.: Fast Algorithms for Maintaining Replica Consistency in Lazy Master Replicated Databases. VLDB, pp. 126–137 (1999)
- Shaheen, A., Gruenwald, L.: Group based replication for mobile ad hoc databases (GBRMAD). Technical Report, University of Oklahoma, Norman, 2000. <http://www.cs.ou.edu/~database/documents/shaheen.pdf>, accessed November 2006
- Tamori, M., Ishihara, S., Watanabe, T., Mizuno, T.: A replica distribution method with consideration of the positions of mobile hosts on wireless ad hoc networks. In: International Conference on Distributed Computing Workshops, pp. 331–335 (2002)
- Wang, K., Li, B.: Efficient and guaranteed service coverage in partitionable mobile ad hoc networks. IEEE INFOCOM, pp. 1089–1098 (2002)